TStringAlignGrid

Unit

AliGrid

Description

Originally it was a descendant of TStringGrid with the possibility to change the alignment of the text to left (same as TStringGrid) or right aligned or centered and only 4k source code, but with each new version more features appeared but the name stayed (as there is already a TSuperGrid I've to stick to the name...), and it now has 200k of source, and there are still further items on my to do list.

For updates and a listing of bugs (often with patches or work-arounds) check my homepage. This software comes as freeware, you may use any way you like. However there is no warranty whatsoever, I can only promise I did my best to avoid bugs. If you want to redistribute this component only do it completely with all the files in the archive. Finally if you like this component all I ask you to do is send me a nice postcard of your hometown - other presents are also welcome but a postcard is enough.

Usage

To install this component select "Install Component" and then select the file `aligridr.pas`. You will then see this component in the "Custom" tab of the component list. Many of the additional properties of this component can already be seen in the object inspector, however the more complex ones (the object inspector does not know about array properties) are only accessible with the component editor - just double click on the component. All the settings are saved as usual in Delphi, but of course you can also access them the Pascal way with the properties as listed in this help.

Advanced Usage

Like every component this one can be extended by inheriting your own custom component from it. I have made preparations in the component to allow to add more functionality in inherited components, and as adding more cell specific items like the fonts will be the most prominent use take a look at the `icongrid.pas`, which add cell specific icons. If you create such a component based upon mine you're welcome to send it to me, so I can either add the functionality in the base class, or at least avoid to become incompatible when doing a new version.

Thanks

A great number of people contributed to this component by reporting bugs, suggesting enhancements or even sending code I just needed to include; many even don't know they contributed to this component when they asked for a special grid component in one of the newsgroups and gave me new ideas this way. So instead of listing those names I can still remember and forgetting many others I just thank everybody who wrote me, and hope you will apologize me if I didn't answered your email...

TStringAlignGrid Properties

TStringAlignGrid

In TStringAlignGrid

Alignment
AlignCell
AlignCol
AlignRow
FixAlignCol
FixAlignRow

CellBrush
ColBrush
RowBrush
FixedColBrush
FixedRowBrush

CellFont
ColFont
RowFont
FixedColFont
FixedRowFont

WordWrap
WordWrapCol
WordWrapRow
WordWrapCell

Editable
EditCell
EditRow
EditCol
SelectEditText
EditMultiline

SelectedCellColor
SelectedColorCell
SelectedColorCol
SelectedColorRow
SelectedFontColor
SelectedFontColorCell
SelectedFontColorCol
SelectedFontColorRow
DrawSelection

HintCell
ShowCellHints

AllowCutnPaste
PasteEditableOnly

RedrawWhileScroll
AutoAdjustLastCol

CellAsDate
CellAsInt

AutoEditNextCell
AfterLastCellEdit
AfterLastCellTab
NextCellEdit
NextCellTab

HTMLCaption
HTMLBorder

TStringAlignGrid Methods

TStringAlignGrid

In TStringAlignGrid

ClearSelection
ShowEdit
HideEdit
ResetHintCellAll
SortColumn
SortRow

RemoveCol
RemoveRow
InsertCol
InsertRow
ExchangeCol
ExchangeRow

AdjustColWidth
AdjustColWidths
AdjustRowHeight
AdjustRowHeights
AdjustLastCol
GetTotalWidth
GetTotalHeight

Contents2HTML
Contents2HTMLClipboard
Contents2CSV
CSV2Contents
Contents2CSVClipboard
ClipboardCSV2Contents
SaveToFile
LoadFromFile
CopyToClipboard
CopyFromClipboard

HorizontalScrollbarVisible
VerticalScrollbarVisible

ResetAlignCell
ResetAlignCellAll
ResetAlignCol
ResetAlignFixedCol
ResetAlignColAll
ResetAlignRow
ResetAlignFixedRow
ResetAlignRowAll

ResetWordWrapCell
ResetWordWrapCellAll
ResetWordWrapCol
ResetWordWrapFixedCol
ResetWordWrapColAll
ResetWordWrapRow
ResetWordWrapFixedRow

ResetWrapRowAll

ResetEditCell
ResetEditCellAll
ResetEditCol
ResetEditColAll
ResetEditRow
ResetEditRowAll

ResetSelectedColorCell
ResetSelectedColorCellAll
ResetSelectedColorCol
ResetSelectedColorColAll
ResetSelectedColorRow
ResetSelectedColorRowAll

ResetSelectedFontColorCell
ResetSelectedFontColorCellAll
ResetSelectedFontColorCol
ResetSelectedFontColorColAll
ResetSelectedFontColorRow
ResetSelectedFontColorRowAll

ResetBrushCell
ResetBrushCellAll
ResetBrushCol
ResetBrushFixedCol
ResetBrushColAll
ResetBrushRow
ResetBrushFixedRow
ResetBrushRowAll

ResetFontCell
ResetFontCellAll
ResetFontCol
ResetFontFixedCol
ResetFontColAll
ResetFontRow
ResetFontFixedRow
ResetFontRowAll

ResetAllCell
ResetAllCellAll
ResetAllCol
ResetAllFixedCol
ResetAllColAll
ResetAllRow
ResetAllFixedRow
ResetAllRowAll

TStringAlignGrid Events

TStringAlignGrid

In TStringAlignGrid

- OnAfterEdit
- OnAfterCancel
- OnBeforeEdit
- OnValidateEdit

- OnColWidthChanged
- OnRowHeightChanged

- OnFixedColClick
- OnFixedRowClick

- OnCompareCol
- OnCompareRow

- OnShowHintCell

Property Hierarchy

If there are several settings which would have an impact on a specific cell they always follow the hierarchy

1. cell
2. column
3. row
4. global

so any cell specific setting is always used, and only if no cell, column and row specific setting is found the global one is used.

Hierarchy

TObject
    |
TPersistent
    |
TComponent
    |
TControl
    |
TWinControl
    |
TCustomControl
    |
TCustomGrid
    |
TDrawGrid
    |
TStringGrid
    |
TStringAlignGrid

TCellEvent type

TCellEvent is used for any event which occurs in a single cell of the grid.

```
type TCellEvent = procedure (Sender: TObject; Col,Row:Longint) of object;
```

T_WordWrap type

See also

Specifies how text longer as the visible width is wrapped.

Unit

aligrid

```
type T_WordWrap = (ww_none, ww_wordwrap, ww_ellipsis, ww_default);
```

Description

The following are possible values of T_WordWrap:

Value	Meaning
ww_none	The text fills the full width, any letters outside will just be cut away.
ww_wordwrap	The text is word wrapped, that means whole words separated by blanks will be moved to further lines. if there are words longer than the available width additional characters will be cut away as in ww_none.
ww_ellipsis	The cut away text is visualized by three trailing dots at the end of the visible area.
ww_Default	Do not override the word wrap and use the one of a lower hierarchy instead

Note

The word wrap style `ww_ellipsis` is only supported for 32 bit Windows, however a 16 bit application as created with Delphi 1 running on a 32 bit Windows can also use this feature. So use this flag with care if you're using Delphi 1.

T_NextCell type

Specifies what the term "next cell" means

Unit

aligrid

```
type T_NextCell = (nc_rightdown, nc_downright, nc_leftdown, nc_downleft,
              nc_leftup, nc_upleft, nc_rightup, nc_upright);
```

Description

The following are possible values of T_NextCell:

Value	Meaning
nc_rightdown	First move to the right, after reaching the end of the line go down. This is a standard column by column editing.
nc_downright	First move down, after reaching the end of the column go one column to the right. This is a standard row by row editing
nc_leftdown	Go to the left, then go down.
nc_downleft	Go down first, then go left.
nc_leftup	Go left first, then go up. This is the opposite direction to nc_rightdown.
nc_upleft	Go up first, then go to the left. This is the opposite to nc_downright.
nc_rightup	Go the the right first, then go up.
nc_upright	Go up first, then go to the right.

T_LastCell type

Specifies what to do after reaching the last cell

Unit

aligrid

```
type T_LastCell = (lc_newcolrow, lc_stop, lc_first, lc_exit);
```

Description

The following are possible values of T_LastCell:

Value	Meaning
lc_newcolrow	A new column or row (depending on the

direction used) will be added to the grid.

lc_stop	Stay in the final cell
lc_first	Start again with the first cell with the current direction.
lc_exit	Gives the focus to the next control in the taborder of the form.

TStringAlignGrid.Alignment

TStringAlignGrid See also

The global alignment of the grid

```
property Alignment: TMyAlign;
```

Description

Set the alignment which is used as a default in the grid. According to the hierarchy of the properties this can be overridden by AlignRow, AlignCol, FixAlignRow, FixAlignCol or AlignCell setting where applicable.

TMyAlign type

See also

Specifies how text is aligned in the cell.

Unit

aligrid

```
type TMyAlign = (alRight,alLeft,alCenter,alDefault);
```

Description

The following are possible values of TMyAlign:

Value	Meaning
alLeft	Text is left-justified: Lines all begin at the left edge of the control.
alRight	Text is right-justified: Lines all end at the right edge of the control.
alCenter	Text is centered in the control.
alDefault	Do not override the alignment and use the one of a lower hierarchy instead

TStringAlignGrid.AlignCell

TStringAlignGrid See also

The alignment for each single cell

```
property AlignCell[ACol,ARow:integer]: TMyAlign;
```

Description

Use this property to set the alignment for each single cell. According to the hierarchy of the properties this overrides any AlignRow, AlignCol, FixAlignRow, FixAlignCol or Alignment setting for the given cell. To retain the alignment of a lower hierarchy set it to `alDefault`, or use the ResetAlignCell method.

TStringAlignGrid.AlignRow

TStringAlignGrid See also

The alignment for a row

```
property AlignRow[ARow:integer]: TMyAlign;
```

Description

Use this property to set the alignment for a row. According to the hierarchy of the properties this overrides the Alignment setting for the given row and may be overridden by a AlignCell setting. To retain the alignment of a lower hierarchy set it to `alDefault`, or use the ResetAlignRow method.

TStringAlignGrid.AlignCol

TStringAlignGrid See also

The alignment for a column

```
property AlignCol[ACol:integer]: TMyAlign;
```

Description

Use this property to set the alignment for a column. According to the hierarchy of the properties this overrides the Alignment setting for the given column and may be overridden by a AlignCell setting. To retain the alignment of a lower hierarchy set it to `alDefault`, or use the ResetAlignCol method.

TStringAlignGrid.FixAlignRow

TStringAlignGrid See also

The alignment for a fixed row

```
property FixAlignRow[ARow:integer]: TMyAlign;
```

Description

Use this property to set the alignment for a fixed row. According to the hierarchy of the properties this overrides the Alignment setting for the given row and may be overridden by a AlignCell setting. To retain the alignment of a lower hierarchy set it to `alDefault`, or use the ResetAlignFixedRow method.

TStringAlignGrid.FixAlignCol

TStringAlignGrid **See also**

The alignment for a fixed column

```
property FixAlignCol[ACol:integer]: TMyAlign;
```

Description

Use this property to set the alignment for a fixed column. According to the hierarchy of the properties this overrides the Alignment setting for the given column and may be overridden by a AlignCell setting. To retain the alignment of a lower hierarchy set it to `alDefault`, or use the ResetAlignFixedCol method.

TStringAlignGrid.ResetAlignCell

TStringAlignGrid See also

Resets the alignment of a given cell back to the default value

```
procedure ResetAlignCell(ACol, ARow: integer);
```

Description

Resets the alignment of a given cell back to the default value according to the hierarchy, it is equivalent to a `AlignCell[ACol,ARow]:=aldefault`. To change all cell specific alignments at once use the ResetAlignCellAll method.

TStringAlignGrid.ResetAlignCellAll

TStringAlignGrid See also

Resets the alignment of a all cells back to the default value

```
procedure ResetAlignCellAll;
```

Description

Resets the alignment of all cells back to the default value according to the hierarchy.

TStringAlignGrid.ResetAlignCol

TStringAlignGrid See also

Resets the alignment of a given column back to the default value

```
procedure ResetAlignCol(ACol: integer);
```

Description

Resets the alignment of a given column back to the default value according to the hierarchy, it is equivalent to a `AlignCol[ACol]:=aldefault`. To reset all column specific alignments at once use the ResetAlignColAll method.

TStringAlignGrid.ResetAlignColAll

TStringAlignGrid See also

Resets the alignment of a all columns (fixed and non-fixed) back to the default value

```
procedure ResetAlignColAll;
```

Description

Resets the alignment of all columns (fixed and non-fixed) back to the default value according to the hierarchy.

TStringAlignGrid.ResetAlignRow

TStringAlignGrid See also

Resets the alignment of a given row back to the default value

```
procedure ResetAlignRow(ARow: integer);
```

Description

Resets the alignment of a given row back to the default value according to the hierarchy, it is equivalent to a `AlignRow[ARow]:=aldefault`. To reset all row specific alignments at once use the ResetAlignRowAll method.

TStringAlignGrid.ResetAlignFixedCol

TStringAlignGrid See also

Resets the alignment of a given fixed column back to the default value

```
procedure ResetAlignFixedCol(ACol: integer);
```

Description

Resets the alignment of a given fixed column back to the default value according to the hierarchy, it is equivalent to a `FixAlignCol[ACol]:=aldefault`. To reset all column specific alignments at once use the ResetAlignColAll method.

TStringAlignGrid.ResetAlignFixedRow

TStringAlignGrid See also

Resets the alignment of a given fixed row back to the default value

```
procedure ResetAlignFixedRow(ARow: integer);
```

Description

Resets the alignment of a given fixed row back to the default value according to the hierarchy, it is equivalent to a `FixAlignRow[ARow]:=aldefault`. To reset all row specific alignments at once use the ResetAlignRowAll method.

TStringAlignGrid.ResetAlignRowAll

TStringAlignGrid See also

Resets the alignment of a all rows (fixed and non-fixed) back to the default value

```
procedure ResetAlignRowAll;
```

Description

Resets the alignment of all rows (fixed and non-fixed) back to the default value according to the hierarchy.

TStringAlignGrid.OnAfterEdit

TStringAlignGrid See also

Occurs after the editing of a cell was finished

```
property OnAfterEdit: TCellEvent;
```

Description

The OnAfterEdit event occurs every time the editing of a cell is finished - by changing to another cell, when the grid looses it's focus, or just by finishing editing by hitting Return.

TStringAlignGrid.OnAfterCancel

TStringAlignGrid See also

The OnAfterCancel event occurs when then editing was canceled.

```
property OnAfterCancel: TCellEvent;
```

Description

The OnAfterCancel event occurs when then editing of a cell was finished by hitting Escape.

TStringAlignGrid.OnBeforeEdit

TStringAlignGrid See also

Occurs before the editing of a cell starts

```
property OnBeforeEdit: TCellEvent;
```

Description

The OnBeforeEdit event occurs every time the editing of a cell begins.

TStringAlignGrid.OnValidateEdit

TStringAlignGrid Example See also

Occurs after the editing of a cell was aborted

```
type TCellEventBool = procedure (Sender: TObject; Col,Row:Longint; var result:
              boolean) of object;
property OnValidateEdit: TCellEventBool;
```

Description

The OnValidateEdit event occurs after the editing of a cell just like the OnAfterEdit event, however this can be used to check the contents of the cell and eventually force the user to keep editing until it is valid.

OnValidateEdit Example

This example uses the OnValidateEdit event to ensure the user only enters valid integer values in the grid.

```
procedure TForm1.StringAligngrid1OnValidateEdit(Sender: TObject; Col,Row:
              Longint; var result: boolean);
var
  dummy: integer;
begin
  try
    dummy:=strtoint(StringAlignGrid1.Cells[Col,Row]);
    result:=true;
  except
    result:=false;
  end;
end;
```

TStringAlignGrid.OnFixedColClick

TStringAlignGrid

Occurs when a fixed column is clicked

```
type TColEvent = procedure (Sender: TObject; Col:Longint) of object;
property OnFixedColClick: TColEvent;
```

Description

The OnFixedColClick event occurs when the user clicks on a fixed column with the mouse.

TStringAlignGrid.OnFixedRowClick

TStringAlignGrid

Occurs when a fixed row is clicked

```
type TRowEvent = procedure (Sender: TObject; Row:Longint) of object;
property OnFixedRowClick: TRowEvent;
```

Description

The OnFixedRowClick event occurs when the user clicks on a fixed row with the mouse.

TStringAlignGrid.OnColWidthChanged

TStringAlignGrid See also

Occurs when the width of a column changes

```
type TColEvent = procedure (Sender: TObject; Col:Longint) of object;
property OnColWidthChanged: TColEvent;
```

Description

The OnColWidthChanged occurs when the width of a column changes, either by user interaction with the mouse or by setting the `ColWidths[Col]` property.

TStringAlignGrid.OnRowHeightChanged

TStringAlignGrid See also

Occurs when the height of a row changes

```
type TRowEvent = procedure (Sender: TObject; Row:Longint) of object;
property OnRowHeightChanged: TRowEvent;
```

Description

The OnRowHeightChanged occurs when the height of a row changes, either by user interaction with the mouse or by setting the `RowHeights[Row]` property.

TStringAlignGrid.HintCell

TStringAlignGrid See also

The hint for each single cell

```
property HintCell[ACol,ARow:integer]: string;
```

Description

Use this property to set a hint specific for each single cell. To switch between the default behavior of showing one hint for the whole component and showing cell hints use the ShowCellHints property. To remove all cell specific hints use the ResetHintCellAll method.

Note

In Delphi 1 and 2 the global event Application.OnShowHint is used internally to make this property work as Borland introduced a more flexible way to access the hints only starting with Delphi 3. If you intend to use the cell specific hints be careful when using this global event, as this is a single method pointer and no array of methods as one would need. In the unit ah_tool you can find some utility functions to allow multiple OnShowHint event handler with Delphi 1 and 2.

TStringAlignGrid.ShowCellHints

TStringAlignGrid See also

Switch the hints between cell specific and grid global

```
property ShowCellHints: boolean;
```

Description

Use this property to enable the cell specific hints set with the HintCell property.

TStringAlignGrid.WordWrap

TStringAlignGrid See also

The global word wrap style of the grid

```
property WordWrap: T_WordWrap;
```

Description

Set the word wrap style which is used as a default in the grid. According to the hierarchy of the properties this can be overridden by WordWrapRow, WordWrapCol or WordWrapCell setting where applicable.

Note

The word wrap style `ww_ellipsis` is only supported for 32 bit Windows, however a 16 bit application as created with Delphi 1 running on a 32 bit Windows can also use this feature. So use this flag with care if you're using Delphi 1.

As with the ellipsis style always the beginning of the text is displayed any alignment setting different from the default left alignment is probably nonsense.

TStringAlignGrid.WordWrapCol

TStringAlignGrid See also

The word wrap style for a column

```
property WordWrapCol[ACol:integer]: T_WordWrap;
```

Description

Set the word wrap style which is used for the given column. According to the hierarchy of the properties this override the WordWrap property and can be overridden by WordWrapCell setting. To retain the word wrap style of a lower hierarchy set it to `ww_Default`, or use the ResetWordWrapCol method.

Note

The word wrap style `ww_ellipsis` is only supported for 32 bit Windows, however a 16 bit application as created with Delphi 1 running on a 32 bit Windows can also use this feature. So use this flag with care if you're using Delphi 1.

As with the ellipsis style always the beginning of the text is displayed any alignment setting different from the default left alignment is probably nonsense.

TStringAlignGrid.WordWrapRow

TStringAlignGrid See also

The word wrap style for a row

```
property WordWrapRow[ARow:integer]: T_WordWrap;
```

Description

Set the word wrap style which is used for the given row. According to the hierarchy of the properties this override the WordWrap property and can be overridden by WordWrapCell setting. To retain the word wrap style of a lower hierarchy set it to `ww_Default`, or use the ResetWordWrapRow method.

Note

The word wrap style `ww_ellipsis` is only supported for 32 bit Windows, however a 16 bit application as created with Delphi 1 running on a 32 bit Windows can also use this feature. So use this flag with care if you're using Delphi 1.

As with the ellipsis style always the beginning of the text is displayed any alignment setting different from the default left alignment is probably nonsense.

TStringAlignGrid.WordWrapCell

TStringAlignGrid See also

The word wrap style for each single cell

```
property WordWrapCell[ACol,ARow:integer]: T_WordWrap;
```

Description

Use this property to set the word wrap style for each single cell. According to the hierarchy of the properties this overrides any WordWrapRow, WordWrapCol or WordWrap setting for the given cell. To retain the alignment of a lower hierarchy set it to `ww_default`, or use the ResetWordWrapCellmethod.

TStringAlignGrid.ResetWordWrapCell

TStringAlignGrid See also

Resets the word wrap style of a given cell back to the default value

```
procedure ResetWordWrapCell(ACol,ARow: integer);
```

Description

Resets the word wrap style of a given cell back to the default value according to the hierarchy, it is equivalent to a `WordWrapCell[ACol.ARow]:=ww_default`. To reset all cell specific word wraps at once use the ResetWordWrapCellAll method.

TStringAlignGrid.ResetWordWrapCol

Resets the word wrap style of a given column back to the default value

```
procedure ResetWordWrapCol(ACol: integer);
```

Description

Resets the word wrap style of a given column back to the default value according to the hierarchy, it is equivalent to a `WordWrapCol[ACol]:=ww_default`. To reset all row specific word wraps at once use the ResetWordWrapColAll method.

TStringAlignGrid.ResetWordWrapRow

TStringAlignGrid See also

Resets the word wrap style of a given row back to the default value

```
procedure ResetWordWrapRow(ARow: integer);
```

Description

Resets the word wrap style of a given row back to the default value according to the hierarchy, it is equivalent to a `WordWrapRow[ARow]:=ww_default`. To reset all row specific word wraps at once use the ResetWordWrapRowAll method.

TStringAlignGrid.ResetWordWrapRowAll

TStringAlignGrid See also

Resets the word wrap of a all rows (fixed and non-fixed) back to the default value

```
procedure ResetWordWrapRowAll;
```

Description

Resets the word wrap style of all rows (fixed and non-fixed) back to the default value according to the hierarchy.

TStringAlignGrid.ResetWordWrapFixedRow

TStringAlignGrid See also

Resets the word wrap style of a given fixed row back to the default value

```
procedure ResetWordWrapFixedRow(ARow: integer);
```

Description

Resets the word wrap style of a given fixed row back to the default value according to the hierarchy, it is equivalent to a `WordWrapFixedRow[ARow]:=ww_default`. To reset all row specific word wraps at once use the ResetWordWrapRowAll method.

TStringAlignGrid.WordWrapFixedCol

TStringAlignGrid See also

The word wrap style for a fixed column

```
property WordWrapFixedCol[ACol:integer]: T_WordWrap;
```

Description

Set the word wrap style which is used for the given fixed column. According to the hierarchy of the properties this override the WordWrap property and can be overridden by WordWrapCell setting. To retain the word wrap style of a lower hierarchy set it to `ww_Default`, or use the ResetWordWrapFixedCol method.

Note

The word wrap style `ww_ellipsis` is only supported for 32 bit Windows, however a 16 bit application as created with Delphi 1 running on a 32 bit Windows can also use this feature. So use this flag with care if you're using Delphi 1.

As with the ellipsis style always the beginning of the text is displayed any alignment setting different from the default left alignment is probably nonsense.

TStringAlignGrid.WordWrapFixedRow

TStringAlignGrid See also

The word wrap style for a fixed row

```
property WordWrapFixedRow[ARow:integer]: T_WordWrap;
```

Description

Set the word wrap style which is used for the given fixed row. According to the hierarchy of the properties this override the WordWrap property and can be overridden by WordWrapCell setting. To retain the word wrap style of a lower hierarchy set it to `ww_Default`, or use the ResetWordWrapFixedRow method.

Note

The word wrap style `ww_ellipsis` is only supported for 32 bit Windows, however a 16 bit application as created with Delphi 1 running on a 32 bit Windows can also use this feature. So use this flag with care if you're using Delphi 1.

As with the ellipsis style always the beginning of the text is displayed any alignment setting different from the default left alignment is probably nonsense.

TStringAlignGrid.ResetWordWrapFixedCol

TStringAlignGrid See also

Resets the word wrap style of a given fixed column back to the default value

```
procedure ResetWordWrapFixedCol(ACol: integer);
```

Description

Resets the word wrap style of a given fixed column back to the default value according to the hierarchy, it is equivalent to a `WordWrapFixedCol[ACol]:=ww_default`. To reset all row specific word wraps at once use the ResetWordWrapColAll method.

TStringAlignGrid.ResetWordWrapColAll

TStringAlignGrid See also

Resets the word wrap of a all columns (fixed and non-fixed) back to the default value

```
procedure ResetWordWrapColAll;
```

Description

Resets the word wrap style of all columns (fixed and non-fixed) back to the default value according to the hierarchy.

TStringAlignGrid.ResetWordWrapCellAll

TStringAlignGrid See also

Resets the word wrap of a all cells back to the default value

```
procedure ResetWordWrapCellAll;
```

Description

Resets the word wrap style of all cells back to the default value according to the hierarchy.

TStringAlignGrid.AllowCutnPaste

TStringAlignGrid

Enable cut and paste of ranges of the grid

```
property AllowCutnPaste: boolean;
```

Description

Ranges of the grid can be copied to and from the clipboard using the standard windows keyboard shortcuts if this property is set to `true`. Can be limited with the PasteEditableOnly property.

TStringAlignGrid.PasteEditableOnly

TStringAlignGrid

Limits the cut and paste of grid ranges to cells which are not read only.

```
property PasteEditableOnly: boolean;
```

Description

Limit the cut and paste of cell ranges to those cells which are not read only. Cells can be made read only using one of the properties EditCell, EditCol, EditRow or Editable.

TStringAlignGrid.EditCell

TStringAlignGrid See also

Makes single cells read only or editable

```
property EditCell[ACol,ARow:integer]: boolean;
```

Description

Use this property to make single cells read only or allow editing for single cells. According to the hierarchy of the properties this overrides any EditRow, EditCol or Editable setting for the given cell. To retain the editable setting of a lower hierarchy the ResetEditCell method.

TStringAlignGrid.EditCol

TStringAlignGrid See also

Makes single columns read only or editable

```
property EditCol[ACol:integer]: boolean;
```

Description

Use this property to make single columns read only or allow editing for single columns. According to the hierarchy of the properties this overrides any Editable setting but will be overridden by a EditCell setting. To retain the editable setting of a lower hierarchy the ResetEditCol method.

TStringAlignGrid.EditRow

TStringAlignGrid See also

Makes single rows read only or editable

```
property EditRow[ARow:integer]: boolean;
```

Description

Use this property to make single rows read only or allow editing for single rows. According to the hierarchy of the properties this overrides any Editable setting but will be overridden by a EditCell setting. To retain the editable setting of a lower hierarchy the ResetEditRow method.

TStringAlignGrid.Editable

TStringAlignGrid See also

The global read only setting of the grid

```
property Editable: boolean;
```

Description

Set the read only setting which is used as a default in the grid. According to the hierarchy of the properties this can be overridden by EditRow, EditCol or EditCell setting where applicable. Note that to have editing enabled you *also* need to set `goEditing` in the grids options.

TStringAlignGrid.ResetEditCell

TStringAlignGrid See also

Resets the editable setting of a given cell back to the default value

```
procedure ResetEditCell(ACol, ARow: integer);
```

Description

Resets the editable setting of a given cell back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetEditCellAll method.

TStringAlignGrid.ResetEditCellAll

TStringAlignGrid See also

Resets the editable setting of a all cells back to the default value

```
procedure ResetEditCellAll;
```

Description

Resets the editable settings of all cells back to the default value according to the hierarchy.

TStringAlignGrid.ResetEditCol

TStringAlignGrid See also

Resets the editable setting of a given column back to the default value

```
procedure ResetEditCol(ACol: integer);
```

Description

Resets the editable setting of a given column back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetEditColAll method.

TStringAlignGrid.ResetEditColAll

TStringAlignGrid See also

Resets the editable setting of a all columns back to the default value

```
procedure ResetEditColAll;
```

Description

Resets the editable settings of all columns back to the default value according to the hierarchy.

TStringAlignGrid.ResetEditRow

TStringAlignGrid See also

Resets the editable setting of a given row back to the default value

```
procedure ResetEditRow(ARow: integer);
```

Description

Resets the editable setting of a given row back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetEditRowAll method.

TStringAlignGrid.ResetEditRowAll

TStringAlignGrid See also

Resets the editable setting of a all rows back to the default value

```
procedure ResetEditRowAll;
```

Description

Resets the editable settings of all rows back to the default value according to the hierarchy.

TStringAlignGrid.SelectEditText

TStringAlignGrid See also

Show the inplace editor with selected text

```
property SelectEditText: boolean;
```

Description

The inplace editor normally shows up with the text selected. By setting this property to `false` the editor will show with no text selected and the cursor at the end of the text.

TStringAlignGrid.EditMultiline

TStringAlignGrid See also

Make the inplace editor multi lined

```
property EditMultiline: boolean;
```

Description

The inplace editor is by default single lined, but can be set to a full multi lined editor with this property.

Note

The same behavior can be accessed by using the `Ctrl-Return` instead of `Return` for a hard line break even in the "single lined" mode. With the multi line mode every the cell cannot be changed with the keyboard alone.

TStringAlignGrid.RedrawWhileScroll

TStringAlignGrid See also

Redraw the grid while the scrollbar is used

```
property RedrawWhileScroll: boolean;
```

Description

Redraw the grid while the scrollbar is used for scrolling in the grid. This may make the scrolling much slower depending on the system and the complexity of the grid.

TStringAlignGrid.ClearSelection

TStringAlignGrid

Makes the grid selection invisible by setting it to an empty range.

```
procedure ClearSelection;
```

Description

Makes the grid selection invisible by setting it to an empty range. This is the same as

```
var
  gridrect: TGridRect;
begin
  gridrect.left:=-1;
  gridrect.right:=-1;
  gridrect.top:=-1;
  gridrect.bottom:=-1;
  stringalignngrid1.selection:=gridrect;
  end;
```

Note

To make the selection permanently invisible you can also use the DrawSelection property.

TStringAlignGrid.HideEdit

TStringAlignGrid See also

Makes the inplace editor invisible.

```
procedure HideEdit(cancel:boolean);
```

Description

Makes the inplace editor visible when it is visible. With the parameter `cancel` both a `return` or an `escape` key can be simulated this way.

TStringAlignGrid.ShowEdit

TStringAlignGrid See also

Makes the inplace editor visible in the current cell if possible.

```
procedure ShowEdit;
```

Description

Makes the inplace editor visible in the current cell, unless the current cell (as defined by the `Col` and `Row` property) is read only or `goEditing` is missing in the grid's options.

TStringAlignGrid.ResetHintCellAll

TStringAlignGrid See also

Removes all cell specific hint texts from the grid.

```
procedure ResetHintCellAll;
```

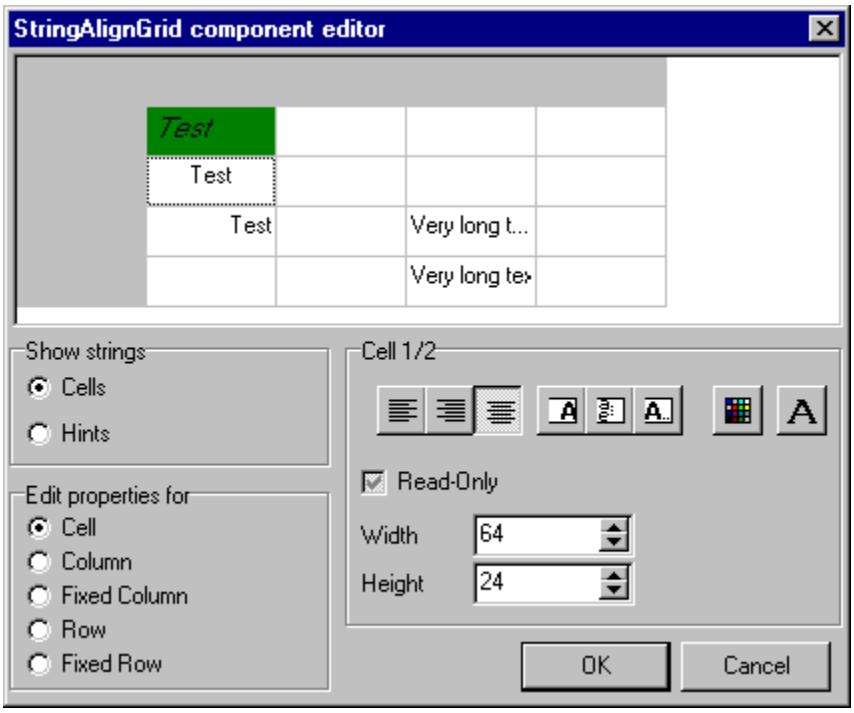
Description

Removes all cell specific hint texts from the grid set with the HintCell property.

Component Editor

TStringAlignGrid

The component editor is very much self explaining. You can toggle between the cell contents or the cell specific hints which you can edit directly in the grid shown, select for what (cell, column, etc.) the settings to the right should apply, and of course set the alignment, word wrap, cell color and font, read only setting and cell size.



The color and font button are down when the current cell (col, row) has a custom color or font set. You can reset this setting by clicking on the button and then select "Cancel" in the color or font selection dialog, then the button will be shown in the up style again.

TStringAlignGrid.OnCompareRow

TStringAlignGrid Example See also

Occurs during the sorting of the grid to provide a custom comparison.

```
type t_relation = (rel_greater, rel_equal, rel_less);
type TCompareFunction = function (Sender: TObject; ColRow, compare1,
                 compare2 :Longint):t_relation of object;
property OnCompareRow: TCompareFunction;
```

Description

During the sorting the comparison of several cells is done using this event, which allows to provide any kind of sorting criteria. There are the predefined comparison functions `CompareColString` and `CompareColInteger` for these standard sorting types. To use on of this two just assign this method to this event

```
StringAlignGrid1.OnCompareRow:=stringgrid1.CompareColString;
```

The parameter `ColRow` gives the row to be sorted (as given in the SortColumn call), and `compare1` and `compare2` contain the two rows to be compared.

TStringAlignGrid.OnCompareCol

TStringAlignGrid Example See also

Occurs during the sorting of the grid to provide a custom comparison.

```
type t_relation = (rel_greater, rel_equal, rel_less);
type TCompareFunction = function (Sender: TObject; ColRow, compare1,
                 compare2 :Longint):t_relation of object;
property OnCompareCol: TCompareFunction;
```

Description

During the sorting the comparison of several cells is done using this event, which allows to provide any kind of sorting criteria. There are the predefined comparison functions `CompareRowString` and `CompareRowInteger` for these standard sorting types. To use on of this two just assign this method to this event

```
StringAlignGrid1.OnCompareCol:=stringgrid1.CompareRowString;
```

The parameter `ColRow` gives the row to be sorted (as given in the SortRow call), and `compare1` and `compare2` contain the two columns to be compared.

TStringAlignGrid.SortRow

TStringAlignGrid Example See also

Sorts the grid with respect to the row given.

```
procedure SortRow(row:longint; ascending:boolean);
```

Description

Sorts the grid with respect to the row given in the `row` parameter. The sorting order is given by the `ascending` parameter, the comparison function to be used must be set assigned to the OnCompareRow event.

TStringAlignGrid.SortColumn

TStringAlignGrid Example See also

Sorts the grid with respect to the column given.

```
procedure SortColumn(column:longint; ascending:boolean);
```

Description

Sorts the grid with respect to the column given in the `column` parameter. The sorting order is given by the `ascending` parameter, the comparison function to be used must be set assigned to the OnCompareCol event.

Sorting Example

This example shows how to use the SortColumn method and OnCompareRow event to soft a grid by interpreting the cell contents in the first column as dates.

```
procedure TForm1.Button1Click(Sender:TObject);
begin
  StringAlignGrid1.SortColumn(0);
  end;

function TForm1.StringAligngrid1OnCompareCol(Sender: TObject;
             ColRow,compare1,compare2: Longint):t_relation;
var
  dummy1,
  dummy2: TDateTime;
begin
  try
    dummy1:=StrToDateTime(    StringAlignGrid1.Cells[colrow,compare1]);
    dummy2:=StrToDateTime(     StringAlignGrid1.Cells[colrow,compare2]);
    if dummy1=dummy2 then
      result:=rel_equal
    else if dummy1>dummy2 then
      result:=rel_greater
    else
      result:=rel_less;
  except
    result:=rel_equal;
  end;
end;
```

TStringAlignGrid.RemoveCol

TStringAlignGrid See also

Delete a single column out of the grid

```
procedure RemoveCol(ACol: integer);
```

Description

Delete a single column out of the grid including all the additional cell parameters.

TStringAlignGrid.RemoveRow

TStringAlignGrid See also

Delete a single row out of the grid

```
procedure RemoveRow(ARow: integer);
```

Description

Delete a single row out of the grid including all the additional cell parameters.

TStringAlignGrid.InsertCol

TStringAlignGrid See also

Insert an empty column at the given position

```
procedure InsertCol(ACol: integer);
```

Description

Insert an empty column into the grid at the given position.

TStringAlignGrid.InsertRow

TStringAlignGrid See also

Insert an empty row at the given position

```
procedure InsertRow(ARow: integer);
```

Description

Insert an empty row into the grid at the given position.

TStringAlignGrid.ExchangeCol

TStringAlignGrid See also

Exchanges two columns in the grid.

```
procedure ExchangeCol(FromIndex, ToIndex: integer);
```

Description

Exchanges the two columns `FromIndex` and `ToIndex` with each other.

TStringAlignGrid.ExchangeRow

TStringAlignGrid See also

Exchanges two rows in the grid.

```
procedure ExchangeRow(FromIndex, ToIndex: integer);
```

Description

Exchanges the two rows `FromIndex` and `ToIndex` with each other.

TStringAlignGrid.AdjustRowHeight

TStringAlignGrid See also

Adjust the height of the row so every cell contents in this row will be displayed properly

```
procedure AdjustRowHeight(ARow:longint);
```

Description

Adjust the height of the row so every cell contents in this row will be displayed properly. The maximum height needed in this row is calculated and set as the height of the row.

TStringAlignGrid.AdjustColWidth

TStringAlignGrid See also

Adjust the width of the column so every cell contents in this column will be displayed properly

```
procedure AdjustColWidth;
```

Description

Adjust the width of the column so every cell contents in this column will be displayed properly. The maximum width needed in this column is calculated and set as the width of the column.

TStringAlignGrid.AdjustRowHeights

TStringAlignGrid See also

Adjust the height of all row so every cell contents in the grid will be displayed properly

```
procedure AdjustRowHeights;
```

Description

Adjust the height of all rows so every cell contents in the grid. It is the same as a call of AdjustRowHeight for every row:

```
for ARow:=StringAlignGrid1.RowCount-1 downto 0 do
  StringAlignGrid1.AdjustRowHeight(ARow);
```

TStringAlignGrid.AdjustColWidths

TStringAlignGrid See also

Adjust the width of all columns so every cell contents in the grid will be displayed properly

```
procedure AdjustColWidths;
```

Description

Adjust the width of all columns so every cell contents in the grid will be displayed properly. It is the same as a call of AdjustColumnWidth for every column:

```
for ACol:=StringAlignGrid1.ColCount-1 downto 0 do
  StringAlignGrid1.AdjustColWidth(ACol);
```

TStringAlignGrid.AdjustLastCol

TStringAlignGrid See also

Adjust the width of the last column to fill the grid entirely.

```
procedure AdjustLastCol;
```

Description

Adjust the width of the last column to fill the grid entirely. This has only an effect if the total width of the columns (as returned with GetTotalWidth) except the last one is less than the ClientWidth of the grid.

TStringAlignGrid.GetTotalWidth

TStringAlignGrid See also

Returns the width allocated by the columns.

```
function GetTotalWidth:longint;
```

Description

Returns the width allocated by the columns including the borders.

TStringAlignGrid.GetTotalHeight

TStringAlignGrid See also

Returns the height allocated by the rows.

```
function GetTotalHeight:longint;
```

Description

Returns the height allocated by the rows including the borders.

TStringAlignGrid.AutoAdjustLastCol

TStringAlignGrid See also

Automatically adjusts the width of the last column when the grid changes size

```
property AutoAdjustLastCol: boolean;
```

Description

Force the grid to call the AdjustLastCol method whenever the size of the grid changes to adjust the last column to fill the grid entirely.

TStringAlignGrid.SelectedCellColor

TStringAlignGrid See also

The background color of the selected cells.

```
property SelectedCellColor: TColor;
```

Description

Change the color used to mark selected cells. According to the hierarchy of the properties this can be overridden by SelectedColorRow, SelectedColorCol or SelectedColorCell setting where applicable. To keep the default behavior of the standard grid set this to `clActiveCaption`.

TStringAlignGrid.SelectedColorRow

TStringAlignGrid See also

The background color of the selected cells for single rows.

```
property SelectedCellRow[ARow: integer]: TColor;
```

Description

Change the color used to mark selected cells for single rows. According to the hierarchy of the properties this can overrides any SelectedCellColor and will be overridden by a SelectedColorCell setting. To retain the color setting of a lower hierarchy the ResetSelectedColorRow method.

TStringAlignGrid.SelectedColorCol

TStringAlignGrid See also

The background color of the selected cells for single columns.

```
property SelectedCellCol[ACol: integer]: TColor;
```

Description

Change the color used to mark selected cells for single columns. According to the hierarchy of the properties this can overrides any SelectedCellColor and will be overridden by a SelectedColorCell setting. To retain the color setting of a lower hierarchy the ResetSelectedColorCol method.

TStringAlignGrid.SelectedColorCell

TStringAlignGrid See also

The background color of the selected cells for single cells.

```
property SelectedCellCell[ACol,ARow: integer]: TColor;
```

Description

Change the color used to mark selected cells for single cells. According to the hierarchy of the properties this can overrides any SelectedColorRow, SelectedColorCol or SelectedCellColor setting. To retain the color setting of a lower hierarchy the ResetSelectedColorCell method.

TStringAlignGrid.DrawSelection

TStringAlignGrid

Switch off the drawing of the selection of the grid.

```
property DrawSelection: boolean;
```

Description

The selection is an often unwanted feature of the grid. Often a call of the ClearSelection method is enough to hide the selection, however with this property it can be made invisible forever.

TStringAlignGrid.ResetSelectedColorCell

TStringAlignGrid See also

Resets the selected color setting of a given cell back to the default value

```
procedure ResetSelectedColorCell(ACol, ARow: integer);
```

Description

Resets the selected color setting of a given cell back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetSelectedColorCellAll method.

TStringAlignGrid.ResetSelectedColorCellAll

TStringAlignGrid See also

Resets all cell specific selected cell colors back to the default value

```
procedure ResetSelectedColorCellAll;
```

Description

Resets all cell specific selected cell colors back to the default value according to the hierarchy.

TStringAlignGrid.ResetSelectedColorCol

TStringAlignGrid See also

Resets the selected color setting of a given column back to the default value

```
procedure ResetSelectedColorCol(ACol: integer);
```

Description

Resets the selected color setting of a given column back to the default value according to the hierarchy.
To change all cell specific editable settings at once use the ResetSelectedColorColAll method.

TStringAlignGrid.ResetSelectedColorColAll

TStringAlignGrid See also

Resets all column specific selected cell colors back to the default value

```
procedure ResetSelectedColorColAll;
```

Description

Resets all column specific selected cell colors back to the default value according to the hierarchy.

TStringAlignGrid.ResetSelectedColorRow

TStringAlignGrid See also

Resets the selected color setting of a given row back to the default value

```
procedure ResetSelectedColorRow(ARow: integer);
```

Description

Resets the selected color setting of a given row back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetSelectedColorRowAll method.

TStringAlignGrid.ResetSelectedColorRowAll

TStringAlignGrid See also

Resets all row specific selected cell colors back to the default value

```
procedure ResetSelectedColorRowAll;
```

Description

Resets all row specific selected cell colors back to the default value according to the hierarchy.

TStringAlignGrid.CellAsDate

TStringAlignGrid

To access the cell contents as a date

```
property CellAsDate[ACol,ARow: integer]: TDateTime;
```

Description

Shortcut to access the cell contents as a date. The conversion will be done using the `DateTimeToStr` and `StrToDateTime` functions. Note that if the cell contents is not a date a read access will raise an exception.

TStringAlignGrid.CellAsInt

TStringAlignGrid

To access the cell contents as an integer

```
property CellAsDate[ACol,ARow: integer]: integer;
```

Description

Shortcut to access the cell contents as an integer. The conversion will be done using the `IntToStr` and `StrToInt` functions. Note that if the cell contents is not an integer a read access will raise an exception.

TStringAlignGrid.AutoEditNextCell

TStringAlignGrid

After finishing editing one cell optionally automatically start the editing in the next one

```
property AutoEditNextCell: boolean;
```

Description

After finishing the editing of a cell by setting this property one can automatically jump into the next (editable) cell for editing. What direction is used to find the next cell can be defined with the NextCellEdit property, so both a column by column or a row by row editing can be achieved. To defined the behavior after finishing the last cell use the AfterLastCellEdit property.

TStringAlignGrid.AfterLastCellTab

TStringAlignGrid

The behavior after the last cell when jumping through the cells with the TAB key.

```
property AfterLastCellTab: T_LastCell;
```

Description

The behavior after the last cell when jumping through the cells with the TAB key with the direction given by the NextCellTab setting.

TStringAlignGrid.NextCellEdit

TStringAlignGrid

The definition of the editing direction for the AutoEditNextCell feature

```
property NextCellEdit: T_NextCell;
```

Description

The definition of the editing direction which is used when then AutoEditNextCell is set. To get a column by column editing use the default value of `nc_rightdown`.

TStringAlignGrid.AfterLastCellEdit

TStringAlignGrid

The behavior after the last cell for the AutoEditNextCell feature

```
property AfterLastCellEdit: T_LastCell;
```

Description

The behavior after reaching the last cell with the AutoEditNextCell feature with the direction given by the NextCellEdit setting.

TStringAlignGrid.NextCellTab

TStringAlignGrid

The definition of the direction used for stepping through the grid with the TAB key

```
property NextCellTab: T_NextCell;
```

Description

The definition of the direction used for stepping through the grid with the TAB key. To go through the grid from the left to the right and then row by row use the standard value of `nc_rightdown`.

TStringAlignGrid.ResetAllCell

TStringAlignGrid See also

Resets all settings of a given cell back to the default value

```
procedure ResetAllCell(ACol, ARow: integer);
```

Description

Resets all settings of a given cell back to the default value according to the hierarchy. To change all cell specific alignments at once use the ResetAllCellAll method.

TStringAlignGrid.ResetAllCellAll

TStringAlignGrid See also

Resets all cell specific settings back to the default value

```
procedure ResetAllCellAll;
```

Description

Resets all cell specific settings back to the default value according to the hierarchy.

TStringAlignGrid.ResetAllCol

TStringAlignGrid See also

Resets all settings of a given column back to the default value

```
procedure ResetAllCol(ACol: integer);
```

Description

Resets all settings of a given column back to the default value according to the hierarchy. To change all column specific alignments at once use the ResetAllColAll method.

TStringAlignGrid.ResetAllColAll

TStringAlignGrid See also

Resets all column specific settings back to the default value

```
procedure ResetAllColAll;
```

Description

Resets all column specific settings back to the default value according to the hierarchy.

TStringAlignGrid.ResetAllFixedCol

TStringAlignGrid See also

Resets all settings of a given fixed column back to the default value

```
procedure ResetAllFixedCol(ACol: integer);
```

Description

Resets all settings of a given fixed column back to the default value according to the hierarchy. To change all column specific alignments at once use the ResetAllColAll method.

TStringAlignGrid.ResetAllRowAll

Resets all row specific settings back to the default value

```
procedure ResetAllRowAll;
```

Description

Resets all row specific settings back to the default value according to the hierarchy.

TStringAlignGrid.ResetAllRow

TStringAlignGrid See also

Resets all settings of a given row back to the default value

```
procedure ResetAllRow(ARow: integer);
```

Description

Resets all settings of a given row back to the default value according to the hierarchy. To change all row specific alignments at once use the ResetAllRowAll method.

TStringAlignGrid.ResetAllFixedRow

TStringAlignGrid See also

Resets all settings of a given fixed row back to the default value

```
procedure ResetAllFixedRow(ARow: integer);
```

Description

Resets all settings of a given fixed row back to the default value according to the hierarchy. To change all row specific alignments at once use the ResetAllRowIAll method.

TStringAlignGrid.CopyToClipboard

TStringAlignGrid See also

Copies the contents of the grid to the clipboard

```
procedure CopyToClipboard;
```

Description

Copies the contents of the grid to the clipboard. The format is the same Excel uses for the clipboard, i.e. separated by TAB characters (ASCII 9). This is a shortcut for the Contents2CSVClipboard call:

```
var
  range: TGridRect;
begin
  range.left:=-1;
  range.right:=-1;
  range.bottom:=-1;
  range.top:=-1;
  Contents2CSVClipboard(#9,range);
  end;
```

TStringAlignGrid.CopyFromClipboard

TStringAlignGrid See also

Copies the contents of the clipboard into the grid

```
procedure CopyFromClipboard;
```

Description

Copies the contents of the clipboard into the grid. The format is the same Excel uses for the clipboard, i.e. separated by TAB characters (ASCII 9). This is a shortcut for the ClipboardCSV2Contents call:

```
var
  range: TGridRect;
begin
  range.left:=-1;
  range.right:=-1;
  range.bottom:=-1;
  range.top:=-1;
  ClipboardCSV2Contents(#9,range);
  end;
```

TStringAlignGrid.LoadFromFile

TStringAlignGrid See also

Loads the contents of the grid from a file

```
procedure LoadFromFile(const filename:string);
```

Description

Loads the contents of the grid from a file, which should have the columns separated by the TAB character (ASCII 9).

TStringAlignGrid.SaveToFile

TStringAlignGrid See also

Copies the contents of the grid to a file

```
procedure SaveToFile(const filename:string);
```

Description

Copies the contents of the grid to a file, the columns separated by the TAB character (ASCII 9).

TStringAlignGrid.Contents2CSVClipboard

TStringAlignGrid See also

Copies the contents of the grid into the clipboard

```
procedure Contents2CSVClipboard(csv: char; range:TGridRect);
```

Description

Copies the contents of the grid to the clipboard. The character to be used for separating the columns is given by the `csv` parameter, the range specifies the area of the grid to be used. A negative value in the range means the extreme value in the given direction, thus a range of `(-1,-1,-1,-1)` will cover the whole grid.

TStringAlignGrid.ClipboardCSV2Contents

TStringAlignGrid See also

Copies the contents of the clipboard into the grid

```
procedure ClipboardCSV2Contents(csv: char; range:TGridRect);
```

Description

Copies the contents of the clipboard into the grid. The character to be used for separating the columns is given by the `csv` parameter, the range specifies the area of the grid to be used. A negative value in the range means the extreme value in the given direction, thus a range of `(-1,-1,-1,-1)` will cover the whole grid.

TStringAlignGrid.CSV2Contents

TStringAlignGrid See also

Copies the contents of a stream into the grid

```
procedure CSV2Contents(data: TStream; csv: char; range:TGridRect);
```

Description

Copies the contents of a stream into the grid. The character to be used for separating the columns is given by the `csv` parameter, the range specifies the area of the grid to be used. A negative value in the range means the extreme value in the given direction, thus a range of `(-1,-1,-1,-1)` will cover the whole grid.

TStringAlignGrid.Contents2CSV

TStringAlignGrid See also

Copies the contents of the grid into the stream

```
function Contents2CSV(data: TMemorystream; csv: char;
              range:TGridRect):TMemorystream;
```

Description

Copies the contents of the grid into a stream. The character to be used for separating the columns is given by the `csv` parameter, the range specifies the area of the grid to be used. A negative value in the range means the extreme value in the given direction, thus a range of `(-1,-1,-1,-1)` will cover the whole grid. To use an already existing stream you can use the `data` parameter, if this is `NIL` a new stream will be created.

TStringAlignGrid.Contents2HTML

TStringAlignGrid See also

Copies the contents of the grid into the stream as a HTML table

```
function Contents2HTML(data: TMemorystream):TMemorystream;
```

Description

Converts the contents of the grid to a HTML table. To use an already existing stream you can use the `data` parameter, if this is `NIL` a new stream will be created. The table style can be modified by the properties HTMLCaption and HTMLBorder.

TStringAlignGrid.Contents2HTMLClipboard

TStringAlignGrid See also

Copies the contents of the grid to the clipboard as a HTML table

```
procedure Contents2HTMLClipboard;
```

Description

Converts the contents of the grid to a HTML table and copies it to the clipboard. The table style can be modified by the properties HTMLCaption and HTMLBorder.

TStringAlignGrid.HTMLCaption

TStringAlignGrid

The caption of the HTML table

```
property HTMLCaption: string;
```

Description

The caption inserted in the table when exported as HTML with Contents2HTML or Contents2HTMLClipboard.

TStringAlignGrid.HTMLBorder

TStringAlignGrid

The border width of the HTML table

```
property HTMLBorder: integer;
```

Description

The border width around the table when exported as HTML with Contents2HTML or Contents2HTMLClipboard.

How to contact

Andreas Hörstemeier
Mefferdatisstraße 16-18
52062 Aachen
Germany

andy@hoerstemeier.de
http://www.hoerstemeier.com

I try to answer as many emails as possible, but as all this programming is done as a hobby please don't be angry if I don't answer promptly - I read all the emails however, and any comment is welcome.

I have created a mailing list which I use to send announcements of new versions of my components, so if you like to get such a notification send an email to ah-delphi-request@scp.de.

Please don't send me questions about Delphi or programming in general, I cannot answer them due to lack of time, you will have much better chances to get an answer by going to the Borland newsgroups at http://www.borland.com/newsgroups or the standard Usenet newsgroups.

TStringAlignGrid.SelectedFontColorRow

TStringAlignGrid See also

The font color of the selected cells for single rows.

```
property SelectedFontColorRow[ARow: integer]: TColor;
```

Description

Change the font color used to mark selected cells for single rows. According to the hierarchy of the properties this can overrides any SelectedFontColor and will be overridden by a SelectedFontColorCell setting. To retain the color setting of a lower hierarchy the ResetSelectedFontColorRow method.

TStringAlignGrid.SelectedFontColorCol

TStringAlignGrid See also

The font color of the selected cells for single columns.

```
property SelectedFontColorCol[ACol: integer]: TColor;
```

Description

Change the font color used to mark selected cells for single columns. According to the hierarchy of the properties this can overrides any SelectedFontColor and will be overridden by a SelectedFontColorCell setting. To retain the color setting of a lower hierarchy the ResetSelectedFontColorCol method.

TStringAlignGrid.SelectedFontColorCell

TStringAlignGrid See also

The font color of the selected cells for single cells.

```
property SelectedFontColorCell[ACol,ARow: integer]: TColor;
```

Description

Change the font color used to mark selected cells for single cells. According to the hierarchy of the properties this can overrides any SelectedFontColorRow, SelectedFontColorCol or SelectedFontColor setting. To retain the color setting of a lower hierarchy the ResetSelectedFontColorCell method.

TStringAlignGrid.SelectedFontColor

TStringAlignGrid See also

The font color of the selected cells.

```
property SelectedFontColor: TColor;
```

Description

Change the font color used to mark selected cells. According to the hierarchy of the properties this can be overridden by SelectedFontColorRow, SelectedFontColorCol or SelectedFontColorCell setting where applicable. To keep the default behavior of the standard grid set this to `clwhite`.

TStringAlignGrid.ResetSelectedFontColorCol

TStringAlignGrid See also

Resets the font color for selected cells of a given column back to the default value

```
procedure ResetSelectedFontColorCol(ACol: integer);
```

Description

Resets the font color of selected cells of a given column back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetSelectedFontColorColAll method.

TStringAlignGrid.ResetSelectedFontColorColAll

TStringAlignGrid See also

Resets all column specific font colors for selected cells back to the default value

```
procedure ResetSelectedFontColorColAll;
```

Description

Resets all column specific font colors for selected cells back to the default value according to the hierarchy.

TStringAlignGrid.ResetSelectedFontColorRow

TStringAlignGrid See also

Resets the font colors for selected cells of a given row back to the default value

```
procedure ResetSelectedFontColorRow(ARow: integer);
```

Description

Resets the font colors of selected cells of a given row back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetSelectedFontColorRowAll method.

TStringAlignGrid.ResetSelectedFontColorRowAll

TStringAlignGrid See also

Resets all row specific font colors for selected cells back to the default value

```
procedure ResetSelectedFontColorRowAll;
```

Description

Resets all row specific font colors for selected cells back to the default value according to the hierarchy.

TStringAlignGrid.ResetSelectedFontColorCell

TStringAlignGrid See also

Resets the font color used for the selected state of a given cell back to the default value

```
procedure ResetSelectedFontColorCell(ACol, ARow: integer);
```

Description

Resets the font color used for the selected state of a given cell back to the default value according to the hierarchy. To change all cell specific editable settings at once use the ResetSelectedFontColorCellAll method.

TStringAlignGrid.ResetSelectedFontColorCellAll

TStringAlignGrid See also

Resets all cell specific font colors of selected cells back to the default value

```
procedure ResetSelectedFontColorCellAll;
```

Description

Resets all cell specific font colors of selected cells back to the default value according to the hierarchy.

TStringAlignGrid.CellBrush

TStringAlignGrid See also

The brush used for the background of a single cell

```
property CellBrush[ACol,ARow: integer]: TBrush;
```

Description

The brush used for the background of a single cell. According to the hierarchy of the properties this overrides any RowBrush, ColBrush, FixedRowBrush, FixedColBrush setting for the given cell. To retain the brush of a lower hierarchy use the ResetBrushCell method.

TStringAlignGrid.ColBrush

TStringAlignGrid See also

The brush used for the background of a single column

```
property ColBrush[ACol: integer]: TBrush;
```

Description

The brush used for the background of a single column. According to the hierarchy of the properties this can be overridden by a CellBrush setting. To retain the brush of a lower hierarchy use the ResetBrushCol method.

TStringAlignGrid.RowBrush

TStringAlignGrid See also

The brush used for the background of a single row

```
property RowBrush[ARow: integer]: TBrush;
```

Description

The brush used for the background of a single row. According to the hierarchy of the properties this can be overridden by a CellBrush setting. To retain the brush of a lower hierarchy use the ResetBrushRow method.

TStringAlignGrid.FixedRowBrush

TStringAlignGrid See also

The brush used for the background of a single fixed row

```
property FixedRowBrush[ARow: integer]: TBrush;
```

Description

The brush used for the background of a single fixed row. According to the hierarchy of the properties this can be overridden by a CellBrush setting. To retain the brush of a lower hierarchy use the ResetBrushFixedRow method.

TStringAlignGrid.FixedColBrush

TStringAlignGrid See also

The brush used for the background of a single fixed column

```
property FixedColBrush[ACol: integer]: TBrush;
```

Description

The brush used for the background of a single fixed column. According to the hierarchy of the properties this can be overridden by a CellBrush setting. To retain the brush of a lower hierarchy use the ResetBrushFixedCol method.

TStringAlignGrid.ResetBrushCell

TStringAlignGrid See also

Resets the brush of a given cell back to the default value

```
procedure ResetBrushCell(ACol, ARow: integer);
```

Description

Resets the brush of a given cell back to the default value according to the hierarchy, it is equivalent to a `CellBrush[ACol,ARow]:=NIL`. To change all cell specific brushs at once use the ResetBrushCellAll method.

TStringAlignGrid.ResetBrushCol

TStringAlignGrid See also

Resets the brush of a given column back to the default value

```
procedure ResetBrushCol(ACol: integer);
```

Description

Resets the brush of a given column back to the default value according to the hierarchy, it is equivalent to a `ColBrush[ACol]:=NIL`. To change all column specific brushs at once use the ResetBrushColAll method.

TStringAlignGrid.ResetBrushRow

TStringAlignGrid See also

Resets the brush of a given row back to the default value

```
procedure ResetBrushRow(ARow: integer);
```

Description

Resets the brush of a given row back to the default value according to the hierarchy, it is equivalent to a `RowBrush[ARow]:=NIL`. To change all row specific brushs at once use the ResetBrushRowAll method.

TStringAlignGrid.ResetBrushCellAll

TStringAlignGrid See also

Resets all cell specific brushs back to the default value

```
procedure ResetBrushCellAll;
```

Description

Resets all cell specific brushs back to the default value according to the hierarchy.

TStringAlignGrid.ResetBrushFixedCol

TStringAlignGrid See also

Resets the brush of a given fixed column back to the default value

```
procedure ResetBrushFixedCol(ACol: integer);
```

Description

Resets the brush of a given fixed column back to the default value according to the hierarchy, it is equivalent to a `FixedColBrush[ACol]:=NIL`. To change all column specific brushs at once use the ResetBrushColAll method.

TStringAlignGrid.ResetBrushFixedRow

TStringAlignGrid See also

Resets the brush of a given fixed row back to the default value

```
procedure ResetBrushFixedRow(ARow: integer);
```

Description

Resets the brush of a given fixed row back to the default value according to the hierarchy, it is equivalent to a `FixedRowBrush[ARow]:=NIL`. To change all row specific brushs at once use the ResetBrushRowAll method.

TStringAlignGrid.ResetBrushColAll

TStringAlignGrid See also

Resets all column specific brushs back to the default value

```
procedure ResetBrushColAll;
```

Description

Resets all column specific brushs back to the default value according to the hierarchy.

TStringAlignGrid.ResetBrushRowAll

TStringAlignGrid See also

Resets all row specific brushs back to the default value

```
procedure ResetBrushRowAll;
```

Description

Resets all row specific brushs back to the default value according to the hierarchy.

Component history

Version	Date	Changes
	1995-12	only a global alignment of the texts in the cells
V1.0	1995-12-17	first published version alignment for cells, columns or global
V1.1	1996-07-03	added reset methods to set the alignments back to the default value
V1.2	1996-12-14	rewrote the internal saving to make it applicable for storing other data added cell specific hints
V1.3	1997-02	not published version added fonts for cells and columns rewrote the Application.OnXXX event handling
V1.4	1997-03-07	added row specific alignment and fonts added component editor for alignments and hints and the component streaming for these new properties
V1.5	1997-05-22	added background colors added the enhanced inplace edit added several utility methods enhanced component editor to show all the array properties
V1.6	1997-09-10	editor functions and events, read only cells import and export methods brushs instead of colors used internally multi lined text support selected cell colors
V1.7	1997-12-07	sorting import and export to the clipboard enhanced the next editable cell functions
V2.0	2000-	redesigned internal storage of the array properties for more flexibility cut and paste cell sizing events and methods more wordwrap options online help compatibility to Delphi 5 and restored full compatibility with Delphi 1

And of course every version fixes bugs of the previous ones, these are not mentioned in this list.

TStringAlignGrid.ColorCell

TStringAlignGrid See also

The background color used for a single cell

```
property ColorCell[ACol,ARow: integer]: TColor;
```

Description

The color used for the background of a single cell. This is the same as CellBrush[ACol,ARow].Color.

Note

This property is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ColorCol

TStringAlignGrid See also

The background color used for a single column

```
property ColorCol[ACol: integer]: TColor;
```

Description

The color used for the background of a single column. This is the same as ColBrush[ACol].Color.

Note

This property is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ColorRow

TStringAlignGrid See also

The background color used for a single row

```
property ColorRow[ARow: integer]: TColor;
```

Description

The color used for the background of a single row. This is the same as RowBrush[ARow].Color.

Note

This property is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.FixColorCol

TStringAlignGrid
See also

The background color used for a single fixed column

```
property FixColorCol[ACol: integer]: TColor;
```

Description

The color used for the background for the fixed cells of a single column. This is the same as FixedColBrush[ACol].Color.

Note

This property is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.FixColorRow

TStringAlignGrid
See also

The background color used for a single fixed row

```
property FixColorRow[ARow: integer]: TColor;
```

Description

The color used for the background for the fixed cells of a single row. This is the same as FixedRowBrush[ARow].Color.

Note

This property is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorCell

TStringAlignGrid See also

Resets the background color of a given cell back to the default value

```
procedure ResetColorCell(ACol, ARow: integer);
```

Description

Resets the background color of a given cell back to the default value according to the hierarchy. This is the same as the ResetBrushCell method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorCellAll

TStringAlignGrid See also

Resets all background colors for single cells to the default value

```
procedure ResetColorCellAll;
```

Description

Resets all cell specific background colors back to the default value according to the hierarchy. This is the same as the ResetBrushCellAll method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorCol

TStringAlignGrid See also

Resets the background color of a given column back to the default value

```
procedure ResetColorCol(ACol: integer);
```

Description

Resets the background color of a given column back to the default value according to the hierarchy. This is the same as the ResetBrushCol method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorRow

TStringAlignGrid See also

Resets the background color of a given row back to the default value

```
procedure ResetColorRow(ARow: integer);
```

Description

Resets the background color of a given row back to the default value according to the hierarchy. This is the same as the ResetBrushRow method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorFixedCol

TStringAlignGrid See also

Resets the background color of a given fixed column back to the default value

```
procedure ResetColorFixedCol(ACol: integer);
```

Description

Resets the background color of a given fixed column back to the default value according to the hierarchy. This is the same as the ResetBrushFixedCol method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorFixedRow

TStringAlignGrid See also

Resets the background color of a given fixed row back to the default value

```
procedure ResetColorFixedRow(ARow: integer);
```

Description

Resets the background color of a given fixed row back to the default value according to the hierarchy. This is the same as the ResetBrushFixedRow method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorColAll

TStringAlignGrid See also

Resets all background colors for single columns to the default value

```
procedure ResetColorColAll;
```

Description

Resets all column specific background colors back to the default value according to the hierarchy. This is the same as the ResetBrushColAll method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.ResetColorRowAll

TStringAlignGrid See also

Resets all background colors for single rows to the default value

```
procedure ResetColorRowAll;
```

Description

Resets all row specific background colors back to the default value according to the hierarchy. This is the same as the ResetBrushRowAll method.

Note

This method is only retained for compatibility with older versions. The brush properties and functions should be used instead.

TStringAlignGrid.CellFont

TStringAlignGrid See also

The font used for the text of a single cell

```
property CellFont[ACol,ARow: integer]: TFont;
```

Description

The font used for the text of a single cell. According to the hierarchy of the properties this overrides any RowFont, ColFont, FixedRowFont, FixedColFont setting for the given cell. To retain the font of a lower hierarchy use the ResetFontCell method.

TStringAlignGrid.ColFont

TStringAlignGrid See also

The font used for the text of a single column

```
property ColFont[ACol: integer]: TFont;
```

Description

The font used for the text of a single column. According to the hierarchy of the properties this can be overridden by a CellFont setting. To retain the font of a lower hierarchy use the ResetFontCol method.

TStringAlignGrid.RowFont

TStringAlignGrid See also

The font used for the text of a single row

```
property RowFont[ARow: integer]: TFont;
```

Description

The font used for the text of a single row. According to the hierarchy of the properties this can be overridden by a CellFont setting. To retain the font of a lower hierarchy use the ResetFontRow method.

TStringAlignGrid.FixedColFont

TStringAlignGrid See also

The font used for the text of a single fixed column

```
property FixedColFont[ACol: integer]: TFont;
```

Description

The font used for the text of a single fixed column. According to the hierarchy of the properties this can be overridden by a CellFont setting. To retain the font of a lower hierarchy use the ResetFontFixedCol method.

TStringAlignGrid.FixedRowFont

TStringAlignGrid See also

The font used for the text of a single fixed row

```
property FixedRowFont[ARow: integer]: TFont;
```

Description

The font used for the text of a single fixed row. According to the hierarchy of the properties this can be overridden by a CellFont setting. To retain the font of a lower hierarchy use the ResetFontFixedRow method.

TStringAlignGrid.ResetFontCell

TStringAlignGrid See also

Resets the font of a given cell back to the default value

```
procedure ResetFontCell(ACol, ARow: integer);
```

Description

Resets the font of a given cell back to the default value according to the hierarchy, it is equivalent to a `CellFont[ACol,ARow]:=NIL`. To change all cell specific fonts at once use the ResetFontCellAll method.

TStringAlignGrid.ResetFontCellAll

TStringAlignGrid See also

Resets all cell specific fonts back to the default value

```
procedure ResetFontCellAll;
```

Description

Resets all cell specific fonts back to the default value according to the hierarchy.

TStringAlignGrid.ResetFontCol

TStringAlignGrid See also

Resets the font of a given column back to the default value

```
procedure ResetFontCol(ACol: integer);
```

Description

Resets the font of a given column back to the default value according to the hierarchy, it is equivalent to a `ColFont[ACol]:=NIL`. To change all column specific fonts at once use the ResetFontColAll method.

TStringAlignGrid.ResetFontColAll

TStringAlignGrid See also

Resets all column specific fonts back to the default value

```
procedure ResetFontColAll;
```

Description

Resets all column specific fonts back to the default value according to the hierarchy.

TStringAlignGrid.ResetFontFixedCol

TStringAlignGrid See also

Resets the font of a given fixed column back to the default value

```
procedure ResetFontFixedCol(ACol: integer);
```

Description

Resets the font of a given fixed column back to the default value according to the hierarchy, it is equivalent to a `FixedColFont[ACol]:=NIL`. To change all column specific fonts at once use the ResetFontColAll method.

TStringAlignGrid.ResetFontFixedRow

TStringAlignGrid See also

Resets the font of a given fixed row back to the default value

```
procedure ResetFontFixedRow(ARow: integer);
```

Description

Resets the font of a given fixed row back to the default value according to the hierarchy, it is equivalent to a `FixedRowFont[ARow]:=NIL`. To change all row specific fonts at once use the ResetFontRowAll method.

TStringAlignGrid.ResetFontRow

TStringAlignGrid See also

Resets the font of a given row back to the default value

```
procedure ResetFontRow(ARow: integer);
```

Description

Resets the font of a given row back to the default value according to the hierarchy, it is equivalent to a `RowFont[ARow]:=NIL`. To change all row specific fonts at once use the ResetFontRowAll method.

TStringAlignGrid.ResetFontRowAll

TStringAlignGrid See also

Resets all row specific fonts back to the default value

```
procedure ResetFontsRowAll;
```

Description

Resets all row specific fonts back to the default value according to the hierarchy.

TStringAlignGrid.OnShowHintCell

TStringAlignGrid Example See also

The OnShowHintCell event occurs when a cell specific hint is about to be displayed.

```
type TShowHintCellProc = procedure (Sender:TObject; col,row:longint; var
              HintStr: string; var CanShow: Boolean; var HintInfo: THintInfo)
              of object;
property OnShowHintCell: TShowHintCellProc;
```

Description

The OnShowHintCell event occurs when a cell specific hint is about to be displayed. Use this event to apply last minute changes to the hint window properties or avoid showing of the hint by setting `CanShow:=false`.

Hint examples

This examples shows how to use the OnShowHint event to provide some alternative behaviour of the cell specific hints. The property ShowCellHints must be set to `true` for this example.

The cell contents itself will be displayed instead of the string set with `HintCell[col,row]`, and the hint window is positioned to cover the cell itself and not below the cell it as usual.

```
procedure TForm1.StringAligngrid1OnShowHintCell(Sender:TObject;
              col,row:longint; var HintStr: string; var CanShow: Boolean; var
              HintInfo: THintInfo);
begin
  hintstr:=(sender as TStringGrid).cells[col,row];
  hintinfo.hintpos.y:=(sender as TStringGrid).Cellrect(col,row).top;
end;
```

TStringAlignGrid.VerticalScrollbarVisible

TStringAlignGrid See also

Returns the visibility of the vertical scrollbar

```
function VerticalScrollbarVisible;
```

Description

Returns the visibility of the vertical scroll bar.

TStringAlignGrid.HorizontalScrollbarVisible

TStringAlignGrid See also

Returns the visibility of the horizontal scrollbar

```
function HorizontalScrollbarVisible;
```

Description

Returns the visibility of the horizontal scroll bar.

Inheriting TStringAlignGrid example

This example shows the basic idea on how to make use of the TStringAlignGrid as the base class to easily add more functionality. There will be an icon added to the cell specific properties, to see the full code take a look at `icongrid.pas` which has everything, only the component editor lacks the icons.

As most extensions will want to make use of the cell specific storage objects this can be inherited

```
TCellPropertiesIcon=class(TCellProperties)
protected
  f_icon: TIcon;
  procedure SetIcon(value: TIcon);
public
  property Icon:TIcon read f_icon write SetIcon;
  destructor destroy;                       override;
  procedure assign(value:TCellProperties); override;
  function isempty: boolean;                override;
  function clone:TCellProperties;           override;
end;
```

Make sure the internal `f_icon` is removed to avoid memory leaks

```
destructor TCellPropertiesIcon.destroy;
begin
  f_icon.free;
  inherited destroy;
  end;
```

The property write method creates the internal `f_icon` if necessary and assigns the image, and if the value is `NIL` the internal icon is removed.

```
procedure TCellPropertiesIcon.SetIcon(value: TIcon);
begin
  if value=NIL then begin
    f_icon.free;
    f_icon:=NIL;
    end
  else begin
    if f_icon=NIL then
      f_icon:=TIcon.Create;
    f_icon.assign(value);
    end;
  end;
```

The assign method just adds the icon to the assignable fields. The type checking is not really necessary as in one grid there should be only one storage class.

```
procedure TCellPropertiesIcon.assign(value:TCellProperties);
begin
  inherited assign(value);
  if value is TCellPropertiesIcon then
    SetIcon(TCellPropertiesIcon(value).icon)
  else
    SetIcon(NIL);
  end;
```

Check if the storage object stores anything, this is used to avoid to store empty fields in the DFM file.

```
function TCellPropertiesIcon.isempty: boolean;
begin
  result:=inherited isempty and ((f_icon=NIL) or (f_icon.handle=0));
```

```
  end;
```

The clone method is a neat utility to make an identical copy of the storage object. This always looks same...

```
function TCellPropertiesIcon.clone:TCellProperties;
begin
  result:=TCellPropertiesIcon.Create(self.f_grid);
  result.assign(self);
  end;
```

Now the grid must have this storage class use as it's storage class which is done by

```
procedure TIconGrid.Initialize;
begin
  inherited Initialize;
  CellPropertiesClass:=TCellPropertiesIcon;
  end;
```

To have a propery CellIcon in the new TIconGrid here's the property stuff.

```
type
  TIconGrid=class(TStringAlignGrid)
    protected
      procedure Initialize;  override;
      procedure IconChanged(AIcon: TObject);
    protected
      function GetIconCell(ACol,ARow: longint):TIcon;
      procedure SetIconCell(ACol,ARow: longint; const Value: TIcon);
    public
      property CellIcon[ACol,ARow:longint]: TIcon read GetIconCell write
               SetIconCell;
    end;

function TIconGrid.GetIconCell(ACol,ARow: longint):TIcon;
var
  v: TCellProperties;
begin
  v:=GetItemCell(ACol,ARow,FPropCell);
  if (v=NIL) then begin
    v:=CellPropertiesClass.Create(self);
    setitemcell(ACol,ARow,FPropCell,v);
    end;
  if TCellPropertiesIcon(v).icon=NIL then begin
    TCellPropertiesIcon(v).f_icon:=TIcon.Create;
    TCellPropertiesIcon(v).icon.OnChange:=iconchanged;
    end;
  result:=TCellPropertiesIcon(v).icon;
  end;

procedure TIconGrid.SetIconCell(ACol,ARow: longint; const Value: TIcon);
begin
  (ObjectCell[ACol,ARow] as TCellPropertiesIcon).icon:=value;
  Invalidate;
  end;
```

This method is needed to make sure the grid is forced to redraw if the icon itself changes by directly

accessing it, e.g. by something like `CellIcon[1,1].Assign(application.icon).`

```
procedure TIconGrid.IconChanged(AIcon: TObject);
begin
  invalidate;
  end;
```

Finally all that is missing is the actual drawing of the icon into the cell. I draw it after drawing the background with the inherited `DrawCellBack`, but before the text itself, and by modifying the `ARect` the text is then limited to the area right to the icon.

```
procedure TIconGrid.DrawCellBack(ACol,ARow:Longint; var ARect:TRect;
              AState:TGridDrawState);
var
  v: TCellProperties;
begin
  inherited DrawCellBack(ACol,ARow,ARect,AState);
  v:=GetItemCell(ACol,ARow,FPropCell);
  if (v<>NIL) and (v is TCellPropertiesIcon) and
              (TCellPropertiesIcon(v).icon<>NIL) and
              (TCellPropertiesIcon(v).icon.handle<>0) then begin
    Canvas.Draw(Arect.Left,Arect.Top,TCellPropertiesIcon(v).icon);
    Arect.Left:=ARect.Left+TCellPropertiesIcon(v).icon.width;
    end;
  end;
```