MultiBase Cosmos

Notes to version 7.6

BASE100

BASE 100, S.A.
www.base100.com

Index

1. Implementations

1.1 APIS

The DLL COSSMTPDLL allows to send emails from a Cosmos program.

E-mails can be sent in both text and HTML format, with the possibility of attaching files in both formats and, in the case of HTML, also include images in the body of the message.

1.2 Runtime

1.2.1 Visual appearance. Cosmos 7

In this versión, new properties have been added in the *Custom Colors* section and new section has been implemented, called *Custom Controls*.

Custom Colors Section

In this section the following characteristics can be defined:

- *The text and the background color of the Panel and Bar controls independently*.
  For this, the following properties have been implemented: Color_Text_Panel, Color_Back_Panel and Color_Back_Bar.
  NOTE: If any of these variables are not defined, the one defined for the Box control will be taken by default.
- *The border color of the Box, Radio, Check, Panel and Bar controls*.
  The new properties are: Color_Border_Box, Color_Border_Radio, Color_Border_Check, Color_Border_Panel, Color_Border_Bar and Color_Border_Text.
- *The background and text color of an edit control when the text is selected*.
  The new properties are: Color_Selected_Text_Edit, Color_Selected_Back_Edit, Color_Selected_Text_DropEdit, Color_Selected_Back_DropEdit, Color_Selected_Text_DropList and Color_Selected_Back_DropList.
- *The border color and text color of an edit control*.
  The new properties are: Color_Border_Edit, Color_Border_DropEdit, Color_Border_DropList, Color_Text_DropEdit, Color_Text_DropEdit, Color_Back_DropList, Color_Text_DropList, Color_Text_Edit and Color_Back_Edit.
- *Transparency in the background color of the controls*.
  If the "background" property is assigned the keyword "*TRANSPARENT*" instead of indicating the RGB value, the controls Will have a transparent background, that is, the wil be displayed over the color defined by the parent control.

To force Cosmos to make the visual aspect of the applications screens the one indicated in the configuration file and not the one indicated in the source code, the VISUALMODEFORZEFORECOLOR and VISUALMODEFORZEBACKCOLOR environment variables have been implemented. If we don't want to show a control with this aspect, we would have to apply the new IgnoreCustomColors method of the SimpleControl class.

Custom Controls Section

This new section allows modifying some characteristics of the Cosmos Form controls at runtime. These characteristics are: aesthetics of the labels, control edges, shape of the corners (rounded or square) and their relief (normal, low or raised).

The appearance of the control Will be modified when any of the following properties are defined, and Will affect all controls in the application.

This section can be defined in the Cosmos configuration file, in the Project configuration file or in an external file that Will be loaded when the new LoadCustomControls method of the Module class is executed. In addition, it Will be necessary to have the COSMOSVISUALMODE variable defined with value 7 and not to use the IgnoreCustomColors method for the control.

What type of controls does it affect?

EDIT, DROPLIST, DROPEDIT, BOX, RADIO, CHECK, TEXT.

What characteristics can be modified?

- **BORDERS.** Indicates which edges of the control will be drawn. If this variable is not defined, all four edges will be drawn. Its possible values are: TOP, BOTTOM, LEFT and RIGHT.

  This property can be defined for EDIT, DROPEDIT, DROPLIST, BOX, RADIO BUTTON, CHECK and TEXT controls.

  Example: EDIT=”BORDERS:BOTTOM,LEFT”

- **CORNERRADIUS.** Indicates the radius in pixels of the circumference of the edge of the controls. If this variable is not defined, the edges will be drawn square. Its value is an integer.

  It only applies if the control frame has the ETCHED property active.

  This property can be defined for BOX, CHECK and RADIO BUTTON type controls.

  Example: BOX="CORNERRADIUS:15"

- **DRAWHEADER.** By defining this property, the area where the label is displayed will have a different background color than the rest of the control. This color Will be the one indicated in the BORDER property of the *Custom Colors* section of the control. The default color is RGB(160,160,160).

  This property can be defined for BOX and RADIO BUTTON type controls.

  Example: BOX="DRAWHEADER"

- **LABELFORECOLOR.** Indicates the color in which the control label text will be drawn.

  This property can be defined for BOX, CHECK and RADIO BUTTON type controls.

  If not defined, the label will be drawn with the color indicated in the *Foreground* property of the control *font*.

  Possible values. Defining color in RGB format using one of these methods:

  a) The expression Rgb(red, green, blue).
  b) With the hexadecimal value that represents the color, using an expression in the following format: #RRGGBB, where RR, GG and BB are the hexadecimal value corresponding to red, green and blue respectively.

Example:
BOX="LABELFORECOLOR:Rgb(128,255,255)"
RADIO:"LABELFORECOLOR:#80FFFF"

- **LABELALIGNMENT.** Indicates the alignment of the control's label text. If this variable is not defined, the label alignment will be to the left. Its possible values are: LEFT, CENTER and RIGHT.

  This property can be defined for BOX and RADIO BUTTON type controls.

  Example: BOX="LABELALIGNMENT:RIGHT"

- **BORDERSTYLE.** Indicates the type of control border to draw for controls of the indicated type. If this variable is not defined, the type of border of the control with which the control will be drawn will be the one indicated at design time. Its possible values are: UP, DOWN and ETCHED.

  This property can be defined for BOX, RADIO BUTTON, CHECK, and TEXT type controls.

  Example: BOX="FRAME:ETCHED"

Properties by control.

EDIT	BORDERS.
DROPEDIT	BORDERS.
DROPLIST	BORDERS.
BOX	BORDERS, CORNERRADIUS, LABELALIGNMENT, DRAWHEADER, LABELFORECOLOR and BORDERSTYLE.
RADIO	BORDERS, CORNERRADIUS, LABELALIGNMENT, DRAWHEADER, LABELFORECOLOR and BORDERSTYLE.
CHECK	BORDERS, CORNERRADIUS and BORDERSTYLE.
TEXT	BORDERS, CORNERRADIUS and BORDERSTYLE.

1.2.2 Screen controls. Defining fonts

From this version on, the font of all controls can be defined in the Cosmos configuration file or in the project configuration file. A font can be assigned for each type of control.

How it works?

To activate this option it is necessary to define the environment variable COSMOSUSECUSTOMFONTS. The value of the fonts will be defined in a new section, called *Custom Fonts*. The type of font assigned to each type of control can be changed if another file is loaded with the LoadCustomFonts method. The call to this method must be made before the Form object is created.

To respect the original Font of the control, the new method IgnoreCustomFonts of the SimpleControl class Will be used. When you need to use a different Font in a certain control, you can use the SetProperty method of the Control class.

Only the properties of the fonts indicated in the file are modified.

Custom Fonts Section

In this section you will define the values of the fonts of the controls. To do this, the following environment variables will be used:

- FONT_BUTTON. Sets the fonts for the BUTTON controls.
- FONT_TEXT. Sets the fonts for the TEXT controls.

- FONT_RADIO_BUTTON_TEXT. Sets the fonts for the RADIO_BUTTON controls.
- FONT_CHECK_TEXT. Sets the fonts for the CHECK controls.
- FONT_BOX. Sets the fonts for the BOX controls.
- FONT_TAB. Sets the fonts for the TABCONTROL controls.
- FONT_PANEL. Sets the fonts for the PANEL controls.
- FONT_BUTTONGROUP. Sets the fonts for the BUTTONGROUP controls.
- FONT_BOXGROUP. Sets the fonts for the BOXGROUP controls.
- FONT_GRID. Sets the fonts for the GRID controls.
- FONT_EDIT. Sets the fonts for the EDITFIELD controls.
- FONT_DROPLIST. Sets the fonts for the DROPLIST controls.
- FONT_DROPEDIT. Sets the fonts for the DROPEDIT controls.
- FONT_MENU. Sets the fonts for the MENU controls.
- FONT_LISTBOX. Sets the fonts for the LISTBOX controls.

1.2.3 List Box Control

Filters

Modifications made to a list filtered with the AllowColumnHeaderFilter method will be reflected in it after removing the filters.

NOTE: This modification does not apply to lists of type Sql.

Alternating colors in lists

It will only show the alternation of colors in the lists when their rows have data. For this, two mechanisms have been implemented:

- AlternateBackColor Method. The RGB value that the method receives is a negative number.
- Define the ALTERNATEBACKCOLORIGNOREEMPTY environment variable.

1.2.4 Grid Control. Events

A new environment variable has been implemented, NOSENDCLICKONGRIDCLICKBODY, to prevent the onClick event from being fired when clicking on the body of the Grid control.

1.2.5 Edit Field Control

The SetEditDateTimePickerIcon method of the class SimpleControl has been implemented, which allows assigning an icon to Edit Field controls of the type DateTimePicker.

1.2.6 Report Viewer

Starting with this version, you can change the background color of the window and view more than one page of the list in the Cosmos report viewer. Full pages will be displayed. The number of pages displayed will depend on their size. The smaller, the more pages.

How it works?

To view more than one page in the Cosmos list viewer, check the “Multi-page” checkbox, located in the lower left part of the window.

When activating Multi-page mode:

- The following controls will be enabled/displayed:

- Checkbox “Show page number”. Checking this box will show the page number of the list in the upper left-hand part of each sheet.
- Zoom buttons (+/-). Increase or decrease the size of the pages.
- Spin control in the lower left margin. Allows you to scroll through the different pages of the list.
  - *First*. If we take the list viewer window as a page, pressing this button will go back one page (it does not move to the first page).
    If the last 12 are displayed in a list of 24 pages, pressing this button will display the first 12.
  - *Previous and Next*. If we divide the window into cells and a page of the list (row, column) is shown in each cell, these buttons allow scrolling through the rows of cells.
    If the first 12 pages of the list are shown in the viewer window, 6 in each row, when pressing the Next button the first page shown in the first cell will be number 7, and in the first cell of the second row it will be number 13.
  - *Last*. If we take the list viewer window as a page, pressing this button will display the next page (it does not scroll to the last page).
    If the first 12 are shown in a list of 24 pages, pressing this button will show the last 12.

- The combination "[Ctrl] + mouse wheel", in addition to zooming, allows you to increase or decrease the number of pages displayed in the viewer.

NOTE: Only full pages will be displayed.

1.2.7 SetSuggestionBox Method

You can define the font to use in the list that shows this method. For this, the SetSuggestionBoxFont method has been implemented.

1.3 Visual Editor

1.3.1 Code Editor

Ability to change the background color and customize the colors of the reserved words.

From this version on, the number will be displayed for each line of code in the left margin.

1.3.2 Project Editor

The possibility of opening the directory where the project source is located has been implemented. For this, a menu option "Open Containing Folder" has been added. This option is in the context menu of the project.

1.3.3 Icons Palette

The ability to search for an icon by name has been implemented.

2. New Methods

2.1 Module Class

2.1.1 LoadCustomControls

This method allows to load the configuration defined in the *Custom Controls* section of the file it receives as a parameter.

Syntax:

```
LoadCustomControls(filePath as Char)
```

Parameter:

FilePath	File full path.

2.1.2 LoadCustomFonts

This method allows to load the configuration defined in the *Custom Fonts* section of the file it receives as a parameter.

Syntax:

```
LoadCustomFonts(filePath as Char)
```

Parameter:

FilePath	File full path.

2.2 PrnDocument Class

2.2.1 SetPreviewBackgroundColor

This method allows you to change the background color of the window shown in the report viewer.

Syntax:

```
SetPreviewBackgroundColor (color as integer)
```

Parameter:

Color	Integer corresponding to the color in RGB format.

2.3 SimpleControl Class

2.3.1 IgnoreCustomFonts

This methods allows you to indicate a particular control Will ignore the properties defined in the *Custom Fonts* section for controls of its type.

Syntax:

```
IgnoreCustomFonts()
```

2.3.2 IgnoreCustomColors

This method allows to indicate that a given control Will ignore the properties defined in the *Custom Colors* and *Custom Controls* sections for controls of its type.

Syntax:

```
IgnoreCustomColors(ignore as Boolean)
```

Parameter:

Ignore	Boolean value indicating whether or not the properties are ignored. This value can be modified in the application as many times as necessary.

2.3.3 SetEditDateTimePickerIcon

This method allows to assign an icon to Edit Field controls of type datetimepiker.

Syntax:

```
SetEditDateTimePickerIcon(iconFile as char, icon as Smallint default 0)
```

Parameters:

iconFile	Icon file identifier.
icon	Index of the icon in the icon file.

2.3.4 SetSuggestionBoxFont

This method allows you to define the font used in the list that is displayed in the SetSuggestionBox method.

Syntax:

```
SetSuggestionBoxFont(Font as char) return Boolean
```

Parameter:

Font	Font to use.

3. Environment variables

ALTERNATEBACKCOLORIGNOREEMPTY

When defining this variable with value TRUE/YES, the colors indicated in the AlternateBackColor method will only be shown in the rows of the List Box control that have data. The background color of the empty rows will be the default color.

Possible values are: TRUE/YES and FALSE/NO.

This variable can be defined in the Environment section of the Cosmos configuration file, the project configuration file or with the PutEnv method of the Module Class.

Its value cannot be modified at runtime.

COSMOSUSECUSTOMFONTS

When defining this variable with TRUE value, the Font assigned to the graphic controls will be the on indicated in the *Custom Fonts* section.

This variable is defined in the Environment section of the Cosmos configuration file, the project's configuration file, or with the PutEnv method of the Module class.

The value of this variable cannot be modified during execution.

NOSENDCLICKONGRIDCLICKBODY

Defining this variable with value TRUE/YES will not trigger the OnClick event when clicking on the body of the Grid.

Possible values are: TRUE/YES and FALSE/NO.

This variable has been implemented to maintain compatibility with version 7.0 of Cosmos (and earlier) in which the OnClick event was not implemented in the client area of the Grid control.

This variable is defined in the Environment section of the Cosmos configuration file, the project's configuration file, or with the PutEnv method of the Module Class.

Its value cannot be modified at runtime.

VISUALMODEFORZEBACKCOLOR

This variable tells Cosmos to force the background color of the controls, even if it is modified during execution, to be the one defined in the *Custom Colors* section of the configuration file.

Possible values are: TRUE/YES and FALSE/NO.

If the value of this variable is FALSE/NO, or if it is not defined, the behavior will be the same as in versions prior to 7.6, that is, the color indicated in design time prevails and not the color defined in the configuration file.

This variable is defined in the Environment section of the Cosmos configuration file, the project's configuration file, or with the PutEnv method of the Module Class.

Its value cannot be modified at runtime.

VISUALMODEFORZEFORECOLOR

This tells Cosmos to force the text color of the controls, even if it is modified during execution, to be the one defined in the *Custom Colors* section of the configuration file.

Possible values are: TRUE/YES and FALSE/NO.

If the value of this variable is FALSE/NO, or if it is not defined, the behavior will be the same as in versions prior to 7.6, that is, the color indicated in design time prevails and not the color defined in the configuration file.

This variable is defined in the Environment section of the Cosmos configuration file, the project's configuration file, or with the PutEnv method of the Module Class.

Its value cannot be modified at runtime.

4. Bug Fixes

4.1 Runtime

- ExportToExcel Method. The font type was not correctly exported to XLSX format.
- Printing to PDF. If in the design of a printing page a Box control was added on top of another, both with a background color, when printing to PDF the background color of the upper box was not painted.
- List Box Control.
    - The ComputeListColumns method did not correctly calculate totals after executing the Sort method.
    - A memory error occurred when the GetColumnSum method was executed on a numeric column (integer) in a LISTBOX control of type string.
- LOOKUPOUTERJOIN. An error occurred when clicking on the header of a column of a Grid that was part of a lookup if the variable was set to TRUE.
- List Box Control. Modifications made to a list filtered with the AllowColumnHeaderFilter method and edited were not saved when removing the filter.
- Grid Control. Clicking the control header did not always fire the OnClick and OnClickHeader events.
- The WriteBinaryEx method did not return the number of bytes written.
- The GETURLFILEEX method did not download some images.
- Grid Control. Clicking on the header of these controls did not always fire the OnClick event or the OnHeaderClick event.
- A memory error occurred and the application stopped working when the GetColumnSum method was executed on a numeric column (integer) in a LISTBOX control of type String.
- List Control Skin. The GAP button did not respect the aesthetics defined in the skin when the mouse was held down on it.
- Cosmos Visual mode 7.
    - It did not correctly draw the border of a button if it had the Highlight attribute. He always drew it in gray.
    - The border of the "no border" and "no label" buttons was not being drawn correctly.