STYLE BY POS

Description

Sets or retrieves the style of a COLUMN, a ROW, or a particular CELL within an edit table.

Applies to

Edit table

Syntax

```
text = Send_Message(controlID, "STYLE_BY_POS", column, row [, style] )
```

Parameters

For **STYLE_BY_POS**, the Send_Message function has the following parameters.

Parameters	Description
column/row	When the column and row parameters are both positive, the style parameter is applied to the cell in the column and row specified. When the column parameter is positive and row parameter is 0, the style parameter is applied to the entire column specified. When the column parameter is 0 and row parameter is positive the style parameter is applied to the entire row specified.
[style]	Optional. When retrieving the current STYLE_BY_POS the style parameter should not be used. When setting the STYLE_BY_POS, the style parameter should be the style to set.

Remarks

The order of processing is CELL overrides ROW, ROW overrides COLUMN.

The Styles that can be set are outlined in the Edit Table Column Styles topic of the Programmer's Reference.

See also

Edit Table Column Styles, COLSTYLE message, DROPDOWNLIST_BY_POS message, COLDROPDOWNLIST message,RTI_Style_Equates

Examples

```
************************************************************

// Flag the first column In an edit table as a drop down column.
// The drop down will allow for the entering of data not in the
// drop down list.
posStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0)
posStyle = bitor(posStyle, DTCS_DROPDOWNEDIT$)
posStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0, posStyle)

// Establish the list of values in the drop down.
dropdownItems = "Gold" : @vm : "Silver" : @vm : "Bronze"

// Populate the Drop Down values In the column
dropDownList = Send_Message(EditTable, "COLDROPDOWNLIST", 1,  dropdownItems)

************************************************************

// Place an Options button in the cell in the third column
// in the first row of an edittable
posStyle = Send_Message(EditTable, "STYLE_BY_POS", 3, 1)
posStyle = bitor(ColStyle,  DTCS_OPTIONSBUTTON$)
```

```
posStyle = Send_Message(EditTable, "STYLE_BY_POS", 3, 1, posStyle)

*****************************************************************

// Create a CheckBox in the Cell at Column 2, Row 5
origColStyle = Send_Message(EditTable, "STYLE_BY_POS", 2,5)
newColStyle = bitor(origColStyle,  DTCS_CHECKBOX$)
ColStyle = Send_Message(EditTable, "STYLE_BY_POS", 2, 5, newColStyle)

// Create a CheckBox across Row 6
origColStyle = Send_Message(EditTable, "STYLE_BY_POS", 0,6)
newColStyle = bitor(origColStyle,  DTCS_CHECKBOX$)
ColStyle = Send_Message(EditTable, "STYLE_BY_POS", 0, 6, newColStyle)

// Create a CheckBox in column 1
origColStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0)
newColStyle = bitor(origColStyle,  DTCS_CHECKBOX$)
ColStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0, newColStyle)

*****************************************************************

// Create a DropDown in the Cell at Column 2, Row 5
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2,5 )
newColStyle = bitor( origColStyle,  DTCS_DROPDOWN$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 5, newColStyle )
dropDownList = "Yes" : @vm : "No"
dropDown = Send_Message( EditTable, "DROPDOWNLIST_BY_POS", 2, 5, dropDownList )

// Create a Drop Down across Row 6
origColStyle = Send_Message(EditTable, "STYLE_BY_POS", 0,6)
newColStyle = bitor(origColStyle,  DTCS_DROPDOWN$)
ColStyle = Send_Message(EditTable, "STYLE_BY_POS", 0, 6, newColStyle)
dropDownList = "Yes" : @vm : "No" : @vm : "Meaningless"
dropDown = Send_Message( EditTable, "DROPDOWNLIST_BY_POS", 0, 6, dropDownList )

// Create a Drop Down in column 1
origColStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0)
newColStyle = bitor(origColStyle,  DTCS_DROPDOWN$)
ColStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0, newColStyle)
dropDownList = "A" : @vm : "B" : @vm : "C"
dropDown = Send_Message( EditTable, "DROPDOWNLIST_BY_POS", 1, 0, dropDownList )

*****************************************************************

// Hide Column 2
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 0  )
newColStyle = bitor( origColStyle,  DTCS_HIDDEN$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 0, newColStyle )

// Unhide Column 2
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 0  )
newColStyle = bitand(origColStyle, bitNot( DTCS_HIDDEN$ ) )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 0, newColStyle )

*****************************************************************

// Protect the  cell  at Column 1, Row 2
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 1, 2  )
newColStyle = bitor( origColStyle,  DTCS_PROTECT$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 1, 2, newColStyle )

// Protect the  all cells within Row 5
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 0, 5  )
newColStyle = bitor( origColStyle,  DTCS_PROTECT$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 0, 5, newColStyle )

*****************************************************************

// Create multiline cells in column 1
colStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0)
colStyle = bitOr(colStyle, DTCS_MULTILINE$)
```

```
colStyle = bitOr(colStyle, DTCS_VSCROLL$)
colStyle = bitOr(colStyle, DTCS_HSCROLL$)
colStyle = bitOr(colStyle, DTCS_AUTOVSCROLL$)
colStyle = bitOr(colStyle, DTCS_AUTOHSCROLL$)
colStyle = Send_Message(EditTable, "STYLE_BY_POS", 1, 0, colStyle)

multilineText  = "This is an example of mulitline text within "
multilineText := "an edit table cell.":char(13):char(10)
multilineText := "The lines are separated with cr/lf.":char(13):char(10)
multilineText := "For maximum results set the RowHeight property "
multilineText := "of the edit table to handle more one line of text."
multilineText :=  char(13) : char(10)

textByPos = Send_Message(EditTable, "TEXT_BY_POS", 1, 1, multilineText)
rowHeight = Send_Message( EditTable, "ROWHEIGHT", 0, 132)

*****************************************************************

// Set the 2nd row to be an uppercase row
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 0, 2  )
newColStyle = bitor( origColStyle,  DTCS_UPPERCASE$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 0, 2, newColStyle )

*****************************************************************

// Vertical Alignment Styles within the Edit Table
// Set the first column's vertical alignment to Center
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 1, 0  )
newColStyle = bitor( origColStyle,  DTCS_VALIGNCENTER$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 1, 0, newColStyle )

// Set the second column's vertical alignment to Bottom
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 0  )
newColStyle = bitor( origColStyle,  DTCS_VALIGNBOTTOM$ )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 2, 0, newColStyle )

// Set the third column's vertical alignment to Top.
// There is not a specific style to set the vertical alignment to Top.
// To set the VALIGN style to TOP it is necessary to remove
// the VALIGNBOTTOM and VALIGNCENTER styles.
origColStyle = Send_Message( EditTable, "STYLE_BY_POS", 3, 0  )
newColStyle = bitAnd( origColStyle, bitNot( DTCS_VALIGNBOTTOM$ ) )
newColStyle = bitAnd( newColStyle, bitNot( DTCS_VALIGNCENTER$) )
ColStyle = Send_Message( EditTable, "STYLE_BY_POS", 3, 0, newColStyle )
* Set the RowHeight in order to confirm the alignment
rowHeight = Send_Message( EditTable, "ROWHEIGHT", 0, 64)
```