Early Morning Editor Control Help

Properties

Events

Ordering

Properties...

Those properties that do not have associated text are standard Visual Basic properties, for information on these properties see the Microsoft Visual Basic Language Reference.

Action
BackColor
BorderStyle
Caption
Count
CaretX
CaretY
DragMode
DragIcon
Enabled
FileOpen
FileSave
FontBold
FontItalic
FontName
FontSize
FontStrike
ForeColor
Height
HelpContextID
hWnd
Index
InsertMode
IsDirty
Left
LeftHi
LinkItem
LinkMode
LinkTimeOut
LinkTopic
MousePointer
Name
Parent
ReadOnly
Redraw
RightHi
ScrollBars
SelDefaultType
SelEndX
SelEndY
SelMark
SelStartX
SelStartY
SelText
TabIndex
TabStop
Tag
Text
TextIndex
Top

TopY
Visible
Width

Events...

Those events that do not have associated text are standard Visual Basic events, for information on these events see the *Microsoft Visual Basic Language Reference*.

BeginMessage
Click
DblClick
DragDrop
DragOver
EndMessage
GotFocus
KeyDown
KeyPress
KeyUp
LinkError
LinkOpen
LinkClose
LinkNotify
LinkChange
LostFocus
MouseDown
MouseMove
MouseUp
ProcessMessage

Ordering...

Unlicensed copies of the Early Morning Editor Control are 100% fully functional so that you can thoroughly evaluate it. Unlicensed copies also have a nag screen that appears at most once whenever the control is loaded into memory and an instance of the control is created, this shouldn't affect your ability to evaluate the control. The registered version of the control is the same as the unregistered version but has no nag screen.

In addition to registered copies of the control, the C++ source code to the control may also be ordered. The source code requires Borland C++ 3.1, Microsoft source is not available. The source code is supplied as is and is not supported in any way. If you are going to order the source code then you should probably also order the registered version of the control but it is not strictly necessary. The control determines if it is registered by looking in the directory from which it was loaded for the presence of a valid license file. The license is only distributed with the registered version of the control, without the registered control you will not be able to use the source without modifying it yourself to disable the nag screens.
In order to build the control from the source you must also have a copy of vbapi.lib (it comes with the Visual Basic Propfessional Edition, for instance).

PRICES...
Registered copies of the Early Morning Editor Control.....$30.
Borland C++ Source Code.....$30.

Shipping and Handling in US and Canada is $4, elsewhere is $6.

CREDIT CARD ORDERS ONLY...
You can order with MC, Visa, Amex, or Discover from Public (software) Library by calling 800-2424-PsL or 713-524-6394 or by FAX to 713-524-6398 or by CIS Email to 71355,470. You can also mail credit card orders to PsL at P.O.Box 35705, Houston, TX 77235-5705.

To insure that you get the latest version, PsL will notify us the day of your order and we will ship the product directly to you.

THE ABOVE NUMBERS ARE FOR ORDERS ONLY.
Any questions about the status of the shipment of the order, refunds, product details, technical support, etc, must be directed to 312-925-1628 or sent to Ted Stockwell, Early Morning Software, 3544 W 83rd Pl., Chicago IL 60652. Returns may be made to this address within 30 days of purchase.

Clipboard

A temporary storage location used to transfer text, graphics, and code.

Insertion Point

The location denoted by the CaretX and CaretY properties. If the control currently has the focus then this is the same as the caret position.

Action Property

Description

This property executes a desired action to be taken. Currently, this property is only used to interface with the Windows clipboard.

Usage

[form.]Editor. Action **= value%**

Remarks

This property can only be assigned. This property is not available at design time.

Setting	Description
0	No Action.
1	Copy any selected text to the Clipboard.
2	Paste a copy of any text on the Clipboard into the text at the insertion point.
3	Remove any selected text and put it on the Clipboard.
4	Delete any selected text.

Data Type

Integer (Enumerated)

Count Property

Description

Specifies the number of lines of text in the control; not available at design time and read-only at run time.

Usage

[form.]Editor.Count

Data Type

Long

CaretX Property

Description

The current X position of the insertion point. The X position of the insertion point may be changed by assigning to this property.
This property is not available at design time.

Usage

[form.]Editor. CaretX**[= value]**

Data Type

Long

See Also

CaretY Property

CaretY Property

Description

The current Y position of the insertion point. The Y position of the insertion point may be changed by assigning to this property.
This property is not available at design time.

Usage

[form.]Editor. CaretY**[= value]**

Data Type

Long

See Also

CaretX Property

SelDefaultType Property

Description

The Early Morning Editor Control supports line, column and stream blocks (stream blocks are the only block type in the standard Visual Basic text control). This property denotes the type of block created when a user interactively marks a block at run-time. Stream blocks are the default.

Usage

[form.]Editor. SelDefaultType**[= value%]**

Setting	Description
0	No blocks allowed.
1	Stream blocks are the default block type.
2	Line blocks are the default block type.
3	Column blocks are the default block type.

Data Type

Integer (Enumerated)

IsDirty Property

Description

Denotes whether the text in the control has changed. The IsDirty property is set to True whenever the text in the control changes (whether the user changes the text or you change the text from code) and remains True until you set the property back to False. The IsDirty property is set to False whenever the FileOpen property is used to load a new file. This property is not available at design time.

Usage

[form.]Editor. IsDirty**[= value%]**

Data Type

Integer (Boolean)

FileOpen Property

Description

Assigning to this property opens a text file and loads it into the control. After a file has been loaded this property contains the full path name of the loaded file. If there is already text in the control when you assign to this property the text will not be saved before the new text is loaded so if you want the text to be saved then you should be sure to save the text beforehand.

Usage

[form.]Editor. FileOpen**[= value%]**

Data Type

String

See Also

FileSave Property

FileSave Property

Description

Assigning to this property saves the text in the control to a file. This property is write-only at run-time.

Usage

[form.]Editor. FileSave **= value%**

Data Type

String

See Also

FileOpen Property

InsertMode Property

Description

When this property is True the control makes room for new text by moving existing text when False the control replaces existing text with incoming text. If you wish this property to be tied to the Insert key then you can trap Insert key presses with the KeyDown event and change the value of the InsertMode property whenever the the Insert key is pressed. The default is to insert text (True).

Usage

[form.]Editor. IsDirty**[= value%]**

Data Type

Integer (Boolean)

Example

This example uses the KeyDown event to trap Insert key presses and flip the InsertMode property...

```
Sub Editor1_KeyDown (KeyCode As Integer, Shift As Integer)
    ' If the Ins key is pressed and neither the Shift, Ctrl, nor Alt
    ' key is down then flip the insert mode property
    If KeyCode = KEY_INSERT And Shift = 0 Then
        Editor1.InsertMode = Not Editor1.InsertMode
    End If
End Sub
```

LeftHi Property

Description

Used to find out what part of a line is marked (highlighted when a block is marked). This property equals the leftmost marked column (1 based) in the line to which the TextIndex refers. If no part of the line is marked then LeftHi equals zero. This property is read-only at run-time and is not available at design time.

Usage

[form.]Editor. LeftHi

Data Type

Long

See Also

RightHi Property, TextIndex Property

ReadOnly Property

Description

Makes the control read only, the user will not be able to alter the text in the control but will be able to scroll through the text, mark blocks, and copy text to the clipboard. The text in the control can still be altered from code, for instance, you can cut text using the Action property when ReadOnly is True. Setting this property to True makes the control read only, the default is False.

Usage

[form.]Editor. ReadOnly**[= value%]**

Data Type

Integer(Boolean)

Redraw Property

Description

Used to reduce the amount of repainting and caret repositioning the control does during a long sequence of operations, thus making the control look like it's operating more smoothly. The default value for this property is True. Set this property to False to disable forced updating and caret repositioning during a long sequence of operations. Be sure to set this property back to True when finished. Not available at design time.

Usage

[form.]Editor. Redraw**[= value%]**

Data Type

Integer(Boolean)

RightHi Property

Description

Used to find out what part of a line is marked (highlighted when a block is marked). This property equals the rightmost marked column (1 based) in the line to which the TextIndex refers. If no part of the line is marked then RightHi equals zero. This property is read-only at run-time and is not available at design time.

Usage

[form.]Editor. RightHi

Data Type

Long

See Also

LeftHi Property, TextIndex Property

ScrollBars Property

Description

Specifies whether an object has horizontal or vertical scroll bars; read-only at run time.

Usage

[form.]Editor. ScrollBars

Setting	Description
0	(Default) None
1	Horizontal
2	Vertical
3	Both

Data Type

Integer (Enumerated)

SelEndX, SelEndY, SelStartX, SelStartY Properties

Description

These four properties denote the starting and ending points of the current block mark, if any, and may be used to size a block mark. These properties are not available at design time.

Usage

[form.]Editor. SelEndX**[= value]**

Data Type

Long

See Also

SelMark Property, SelDefaultType Property

Example

The following example marks all the text without moving the caret...

```
Editor1.SelMark = 0 'unmark any existing block mark, if any
Editor.SelMark = 2 ' start a line block
Editor1.SelStartX = 1
Editor1.SelStartY = 1
Editor1.SelEndX = 1
Editor1.SelEndY = 1
```

SelMark Property

Description

Denotes the type of any current block mark. Can also be used to start a block mark or change the type of block. Not available at design time. The default block type is the block type specified by the SelDefaultType property.

Usage

[form.]Editor. SelMark**[= value]**

Setting	Description
0	No block.
1	Stream block.
2	Line blocks.
3	Column block.

Data Type

Integer (Enumerated)

See Also

SelEndX Property, SelDefaultType Property

SelText Property

Description

When read this property returns the highlighted text in the line to which the TextIndex property refers. When assigned this property replaces any current block mark with the given text, if there is no block mark then the given text is inserted at the insertion point. This property is not available at design time.

Usage

[form.]Editor. SelText**[= value]**

Data Type

String

See Also

SelMark Property, SelDefaultType Property
, TextIndex Property

TextIndex Property

Description

Used to refer to a line of text in the control. This property is used by other properties too determine what line to act on. The lines of text in a control are numbered beginning with one. This property is not available at design time.

Usage

[form.]Editor. TextIndex**[= value]**

Data Type

Long

Text Property

Description

When read this property returns the text in the line to which the TextIndex property refers. When assigned this property replaces the text in the line to which the TextIndex property refers with the given text. This property is not available at design time.

Usage

[form.]Editor. Text**[= value]**

Data Type

String

See Also

, TextIndex Property

TopY Property

Description

Denotes the line displayed at the top of the control's window. The line dispalyed at the top may be changed by assigning to this property. This property is not available at design time.

Usage

[form.]Editor. TopY**[= value]**

Data Type

Long

BeginMessage Event, EndMessage Event, ProcessMessage Event

Description

These are general purpose events that can be used to trap and process any message coming to the control. They are also good for detecting when some unique kind of event happens. BeginMessage is sent when a message is received and EndMessage is sent after the control has finished whatever processing it does, if any, in response to the message. The fProcessMessage parameter to the BeginMessage event can be used to disable the control's response to a message. If the fProcessMessage parameter is set to False then the control will not process the message, instead the ProcessMessage event is fired where you may write your own code to process the message. The control does not fire additional BeginMessage and EndMessage events in response to messages received during the processing of a BeginMessage, EndMessage, or ProcessMessage event.

Syntax

Sub Editor1_BeginMessage (HControl As Long, HWindow As Integer, Message As Integer, WParam As Integer, LParam As Long, fProcessMessage as Integer)

Sub Editor1_EndMessage (HControl As Long, HWindow As Integer, Message As Integer, WParam As Integer, LParam As Long)

Sub Editor1_ProcessMessage (HControl As Long, HWindow As Integer, Message As Integer, WParam As Integer, LParam As Long, MessageResult As Long)

Remarks

If fProcessMessage is set to False during the processing of a BeginMessage event then the control will not perform the processing that it would normally perfrom in response to the message.