Help for List/X+

Properties Events Methods Objects Interfaces

How To Buy This Software

Order Form

Getting Custom Controls Written

Licensing Information

Description

List/X+ is an ActiveX control that sets a new standard for functionality and programmability in ActiveX controls. List/X+ provides multiple sortable columns, multi-line headers, captions, and list items, unlimited color options, and more. The Mabry List/X+ control is a 32-bit light ActiveX.

List/X+ also provides a unique owner-draw capability using high-level COM objects that you create with Visual Basic. This feature provides unlimited flexibility, freeing you from the design limitations built into other controls. Using owner-draw objects, you can easily define the exact appearance of any column in any row. With List/X+ you can display bitmaps, change fonts and colors on the fly, and use any of the graphical methods available in Visual Basic or the Windows API.

FUNCTIONALITY

* Multiple columns

* Multi-column sorting -- individual columns can be sorted as desired

* Sorted/unsorted property may be modified at runtime

* Multi-line list items -- the individual items can be single line or multi-line

* Multi-line column headings -- each column can have its own multi-line heading

* Multi-line caption -- captions are not limited to single lines.

* Unique and innovative COM callbacks provide the programmer with the ability to paint any or all listbox items on a per-column basis

* ItemData is a Variant instead of a Long -- you are no longer limited to numbers only as the data type. The list items can be associated with numeric values, objects, arrays, boolean values, or any other value that can be held in a Variant.

* Simple selection, multiselect, and extended multiselect

* Selection-type is programmable at run-time

* Column alignment may be modified at run-time

* User resizable columns

* Minimum and maximum widths for user-resizable columns

* Properties to determine whether the mouse is over the caption, headings, or list parts of the listbox

* Powerful object-oriented programming model with Column objects and Columns collection

* ColumnIndex property provides the column location of the mouse

* List items are assignable at design-time using property pages

APPEARANCE

* 3D effects for list, caption, and headings include Raised, Pop, Drop, Shadow, or Inset

* Programmable 3D effect colors

* Caption is alignable to left, right, or center

* Selected and normal foreground and background colors may be specified for the listbox, columns, and individual items

* Appearance and colors are programmable at runtime

* Column alignment right, left, or center -- each column can have its own alignment setting

* Column headings alignment right, left or center -- each column heading can have its own alignment

File Name

MLISTX.OCX

ActiveX / OCX Object Name

Mabry.MListCtl

ActiveX Compatibility

VB 4.0 (32-bit), 5.0 and 6.0

ActiveX Built With

Microsoft Visual C++ v5

ActiveX - Required DLLs

None. This is a light ActiveX control that requires no extra DLLs to operate. This means that your installation packages will be smaller. And, your web pages will load faster, since there are no extra DLLs to download.

Distribution Note When you develop and distribute an application that uses this control, you should install the control file into the user's Windows SYSTEM directory. The control file has version information built into it. So, during installation, you should ensure that you are not overwriting a newer version.

Close

List/X+ Properties

Properties that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*BackColor** Property

*Caption** Property

*CaptionAlignment** Property

*CaptionBorderEffect** Property

*CaptionPaintObject** Property

*ColDelimiter** Property

*ColRowOrder** Property

*ColumnIndex** Property

*Columns** Property

*Font** Property

*ForeColor** Property

*HeadingsBorderEffect** Property

*HiliteColor** Property

*HiliteTextColor** Property

*HorzDivForeColor** Property

*HorzDividers** Property

hWnd Property

*ItemBackColor** Property

*ItemData** Property

*ItemForeColor** Property

*ItemHiliteColor** Property

*ItemHiliteTextColor** Property

*List** Property

*ListBorderEffect** Property

*ListCount** Property

*ListIndex** Property

*MouseOverCaption** Property

*MouseOverHeading** Property

*MouseOverList** Property

*MousePointer** Property

*MultiSelect** Property

*Sorted** Property

*SortOrder** Property

*TabStop** Property

*ThreedDarkShadowColor** Property

*ThreedFaceColor** Property

*ThreedHiliteColor** Property

*ThreedShadowColor** Property

*TopIndex** Property

*Version** Property

*VertDivForeColor** Property
*VertDividers** Property

Close

List/X+ Events

Events that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

Click Event

DblClick Event

KeyDown Event

KeyPress Event

KeyUp Event

MouseDown Event

MouseMove Event

MouseUp Event

***RClick** Event

Close

List/X+ Methods

Methods that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__About__ Method

*__AddItem__ Method

*__Clear__ Method

*__PutItems__ Method

*__RemoveItem__ Method

*__ReplaceItem__ Method

Close

List/X+ Objects

Objects that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__Column__ Object

*__ColumnsCollection__ Object

*__MPicture__ Object

List/X+ Interfaces

Interfaces that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__IMCaptionPaint__ Interface
*__IMListPaint__ Interface
*__IMSort__ Interface

Close

List/X+ Column Object Properties

Properties that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__BackColor__ Property
*__ColumnAlignment__ Property
*__ForeColor__ Property
*__HeaderDisplayable__ Property
*__Heading__ Property
*__HeadingAlignment__ Property
*__HiliteColor__ Property
*__HiliteTextColor__ Property
*__MaxWidth__ Property
*__MinWidth__ Property
*__PaintObject__ Property
*__SortObject__ Property
*__UserResizeEnabled__ Property
*__Visible__ Property
*__Width__ Property

Close

List/X+ ColumnsCollection Object Properties

Properties that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__Count__ Property

*__Item__ Property

Close

List/X+ ColumnsCollection Object Methods

Methods that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__Add__ Method

*__Remove__ Method

Close

List/X+ IMCaptionPaint Interface Events

Events that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__CaptionHeight__ Event

*__PaintCaption__ Event

Close

List/X+ IMListPaint Interface Events

Events that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__ItemHeight__ Event

*__PaintColumn__ Event

Close

List/X+ IMSort Interface Events

Events that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__CompareColumns__ Event

Close

List/X+ MPicture Object Properties

Properties that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__BackColor__ Property
*__CurrentX__ Property
*__CurrentY__ Property
*__DrawStyle__ Property
*__DrawWidth__ Property
*__FillColor__ Property
*__FillStyle__ Property
*__Font__ Property
*__FontTransparent__ Property
*__ForeColor__ Property
*__hDC__ Property
*__Picture__ Property
*__ScaleHeight__ Property
*__ScaleMode__ Property
*__ScaleWidth__ Property

Close

List/X+ MPicture Object Methods

Methods that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

- *__Cls__ Method
- *__DrawText__ Method
- *__Line__ Method
- *__PaintPicture__ Method
- *__Point__ Method
- *__PSet__ Method
- *__TextHeight__ Method
- *__TextWidth__ Method

How To Buy This Software

CREDITS

List/X+ was written by Zane Thomas.

CONTACT INFORMATION

Orders, inquiries, technical support, questions, comments, etc. can be sent to mabry@mabry.com on the Internet. Our mailing address/contact information is:

Mabry Software, Inc.
503 316th Street Northwest
Stanwood, WA 98292

Sales: 1-800-99-MABRY (U.S. Only)
Voice: 360-629-9278
Fax: 360-629-9278
Web: http://www.mabry.com

COST

The price of List/X+ (control only) is US$65 (US$70 for International orders). The cost of List/X+ and the C/C++ source code (of the control itself) is US$249 (US$254 for International orders).

Prices are subject to change without notice.

Printed manuals are available at US$12.50 per copy.

DELIVERY METHODS

We can ship this software to you via air mail and/or e-mail.

Air Mail - you will receive diskettes, a printed manual (if purchased), and printed receipt if you choose this delivery method. The costs are:

US$10.00	US Priority Mail
US$15.00	Airborne Express 2nd Day (US deliveries only)
US$20.00	Airborne Express Overnight (US deliveries only)
US$20.00	Global Priority Mail (Int'l deliveries only; Western Europe, Pacific Rim and Canada only)
US$45.00	International Airborne Express (Int'l deliveries only)

E-Mail - We can ship this package to you via e-mail. You need to have an e-mail account that can accept large file attachments (which includes CompuServe, AOL, and most Internet providers). We will e-mail a receipt to you.

Be sure to include your full mailing address with your order. Sometimes (on the Internet) the package cannot be e-mailed, so we are forced to send it through the normal mails.

CompuServe E-Mail - CompuServe members can use the software registration forum (GO SWREG) to register this package. List/X+'s SWREG ID number is 16204. The source code version's ID number is 16205. PLEASE NOTE: When you order through SWREG, we send the registered package to your CompuServe account (not your Internet or AOL account) within a few hours.

ORDER / PAYMENT METHODS

You can order this software by phone, fax, e-mail, mail. For your convenience, an order form has been provided that you can print out directly from this help file.

Please note that orders must include all information that is requested on our order form. Your shipment WILL BE DELAYED if we have to contact you for additional information (such as phone number, street address, etc.).

You can pay by credit card (VISA, MasterCard, American Express, Discover, NOVUS), check (U.S. dollars drawn on a U.S. bank), cash, International Money Order, International Postal Order, Purchase Order (established business entities only - terms net 30), or wire transfer.

WIRE TRANSFER INFORMATION

Here is the information you need regarding our account for a wire funds transfer:

Bank Name:	SeaFirst - Stone Way Branch
Bank Address:	3601 Stone Way North Seattle, WA 98103
Bank Phone:	206-585-4951
Account Name:	Mabry Software, Inc.
Routing Number:	12000024
Account Number:	16311706

If you are paying with a wire transfer of funds, please add US$25.00 to your order. This is the fee that SeaFirst Bank charges Mabry Software. Also, please ADD ANY ADDITIONAL FEES THAT YOUR BANK MAY CHARGE for wire transfer service. If you are paying with a wire transfer, we must have full payment deposited to our account before we can ship your order.

Close **List/X+ Order Form**

Use the Print Topic... command from the File menu to print this order form.

Mail this form to:	Mabry Software, Inc. 503 316th Street Northwest Stanwood, WA 98292
	Phone: 360-629-9278 Fax: 360-629-9278 Internet: mabry@mabry.com Web: www.mabry.com

Where did you get this copy of List/X+?

____

Name: ____

Ship to: ____

____

____

____

Phone: ____

Fax: ____

E-Mail: ____

Credit Card #: ________________________________ exp. ______________

P.O. # (if any): ________________ Signature ____________________________

qty ordered	____	REGISTRATION $65.00 ($70.00 international). Check or money order in U.S. currency drawn on a U.S. bank. Add $10.00 per order for shipping and handling. Add $12.50 per printed manual.
qty ordered	____	SOURCE CODE AND REGISTRATION $249.00 ($254.00 international). Check or money order in U.S. currency drawn on a U.S. bank. Add $10.00 per order for shipping and handling. Add $12.50 per printed manual.

See Also

ColDelimiter Property

ListCount Property

See Also

ForeColor Property

HiliteColor Property

HiliteTextColor Property

See Also

CaptionAlignment Property

CaptionBorderEffect Property

See Also

ListBorderEffect Property

HeadingsBorderEffect Property

See Also

IMCaptionPaint CaptionHeight Event

IMCaptionPaint PaintCaption Event

See Also

RemoveItem Method

See Also

Item Property

See Also

PutItems Method

See Also

MouseOverCaption Property

MouseOverHeading Property

MouseOverList Property

See Also

ColumnsCollection Count Property

ColumnsCollection Add Method

ColumnsCollection Remove Method

ColumnsCollection Item Property

See Also

BackColor Property

HiliteColor Property

HiliteTextColor Property

See Also

CaptionBorderEffect Property

ListBorderEffect Property

See Also

HiliteTextColor Property

See Also

HiliteColor Property

See Also

HorzDividers Property

VertDivForeColor Property

See Also

ItemForeColor Property

ItemHiliteColor Property

ItemHiliteTextColor Property

See Also

ListCount Property

See Also

ItemBackColor Property

ItemHiliteColor Property

ItemHiliteTextColor Property

See Also

ItemForeColor Property

ItemHiliteColor Property

ItemHiliteTextColor Property

See Also

ItemForeColor Property

ItemHiliteColor Property

ItemHiliteTextColor Property

See Also

AddItem Method

ReplaceItem Method

See Also

CaptionBorderEffect Property

HeadingsBorderEffect Property

See Also

MouseOverHeading Property

MouseOverList Property

See Also

MouseOverCaption Property

MouseOverList Property

See Also

MouseOverCaption Property

MouseOverHeading Property

See Also

ColRowOrder Property

See Also

ListCount Property

AddItem Method

ReplaceItem Method

See Also

AddItem Method

See Also

SortOrder Property

IMSort CompareColumns Event

See Also

Sorted Property

IMSort CompareColumns Event

See Also

ThreedShadowColor Property

ThreedFaceColor Property

ThreedHiliteColor Property

See Also

ThreedDarkShadowColor Property

ThreedShadowColor Property

ThreedHiliteColor Property

See Also

ThreedDarkShadowColor Property

ThreedShadowColor Property

ThreedFaceColor Property

See Also

ThreedDarkShadowColor Property

ThreedFaceColor Property

ThreedHiliteColor Property

See Also

HorzDivForeColor Property

VertDividers Property

See Also

ForeColor Property

HiliteColor Property

HiliteTextColor Property

See Also

BackColor Property

HiliteColor Property

HiliteTextColor Property

See Also

Heading Property

Visible Property

Width Property

See Also

HeaderDisplayable Property

HeadingAlignment Property

Visible Property

Width Property

See Also

ColumnAlignment Property

Heading Property

See Also

ForeColor Property

BackColor Property

HiliteTextColor Property

See Also

ForeColor Property

BackColor Property

HiliteColor Property

See Also

MinWidth Property

UserResizeEnabled Property

See Also

MaxWidth Property

UserResizeEnabled Property

See Also

MPicture Picture Property

See Also

IMSort CompareColumns Event

See Also

MaxWidth Property

MinWidth Property

See Also

HeaderDisplayable Property

Width Property

See Also

HeaderDisplayable Property

MaxWidth Property

MinWidth Property

UserResizeEnabled Property

See Also

ColumnsCollection Count Property

ColumnsCollection Item Property

ColumnsCollection Remove Method

See Also

ColumnsCollection Add Method

ColumnsCollection Remove Method

See Also

ColumnsCollection Count Property

See Also

ColumnsCollection Add Method

ColumnsCollection Count Property

See Also

IMCaptionPaint PaintCaption Event

See Also

IMListPaint PaintColumn Event

See Also

ForeColor Property

Cls Method

See Also

BackColor Property

See Also

MPicture CurrentY Property

See Also

MPicture CurrentX Property

See Also

MPicture DrawWidth Property

See Also

MPicture DrawStyle Property

See Also

MPicture FillStyle Property

See Also

MPicture FillColor Property

See Also

MPicture ForeColor Property

MPicture DrawText Method

See Also

MPicture BackColor Property

See Also

MPicture PaintPicture Method

See Also

MPicture PSet Method

See Also

MPicture Point Method

See Also

MPicture ScaleMode Property

MPicture ScaleWidth Property

See Also

MPicture ScaleHeight Property

MPicture ScaleWidth Property

See Also

MPicture TextWidth Method

MPicture Font Property

See Also

MPicture TextHeight Method

MPicture Font Property

About Method

Description

Displays About Box.

Syntax

object.**About**

The syntax of the **About** method has these parts:

Part	Description
object	Required. A List/X+ control.

ColumnsCollection Object Add Method

See Also

Description

Adds one column and returns a reference to it.

Syntax

object.**Add***colindex*

The syntax of the **Add** method has these parts:

Part	Description
object	Required. A ColumnsCollection object.
colindex	Required. An integer specifying the position for the new column.

Remarks

Adds a new column.   The index argument specifies how the column will be ordered relative to other columns. An index of zero is used either to add the first column or to place a column first.   An index of ColumnsCollection.Count adds a column to the end (rightmost column) of the collection.   An index of one inserts a column after the first column and before the second, if it exists.

The Add method returns a reference to the newly added Column object.

AddItem Method

See Also

Description

Adds an item to the listbox.

Syntax

object.**AddItem***item*,*index*

The syntax of the **AddItem** method has these parts:

Part	Description
object	Required. A List/X+ control.
item	Required. A string expression specifying a new item to add to the list.
index	Optional. A long integer specifying the position for the new item.

Remarks

If the listbox has more than one column you should supply a string for each column by concatenating the column strings together, separated by ColDelimiter.

index may be any value between 0 and ListCount - 1.

Column Object BackColor Property

See Also

Description

Returns/sets the column's background color.

Syntax

object.**BackColor** [= *color*]

The syntax of the **BackColor** property has these parts:

Part	Description
object	Required. A Column object.
color	A color expression that specifies the background color of this column.

Remarks

Specifies the background color for a column. The listbox itself and individual list items also have foreground color properties. When a column for some item is about to be displayed, the item's BackColor property is checked. If it has been set then that color is used. If the item's BackColor has not been set, then the column's BackColor is used if it has been set. Otherwise, the listbox's BackColor is used.

Data Type

Color

MPicture Object BackColor Property

See Also

Description

Returns/sets the picture object's BackColor.

Syntax

object.**BackColor** [= *color*]

The syntax of the **BackColor** property has these parts:

Part	Description
object	Required. An MPicture object.
color	A color expression that specifies the picture's background color.

Remarks

Specifies the color used with graphics methods that use a background color.

Data Type

Color

BackColor Property

See Also

Description

Determines the color used for the text in the control.

Syntax

object.**BackColor** [= *color*]

The syntax of the **BackColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression specifying the background color of the control.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different backcolor properties.

Each list item has ForeColor and BackColor properties. If the BackColor for a given item has been specified, then that color is used as the BackColor for each of that item's columns.

Also each Column has a BackColor property. If a value other than -1 is assigned to a Column's BackColor property and the Item's BackColor property is -1, then the color specified by the Column's BackColor property is used.

If neither of the Column or Item BackColors has been specified, then the listbox's BackColor property is used when displaying a given row/column.

Any number of color combinations can be used, if you draw items yourself, by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

Caption Property

See Also

Description

Returns/sets the listbox caption.

Syntax

object.**Caption** [= *string*]

The syntax of the **Caption** property has these parts:

Part	**Description**
object	A List/X+ control.
string	A string expression specifying the control's caption.

Remarks

Multiline captions can be created by including one or more chr$(13) characters in the caption string.

Data Type

String

CaptionAlignment Property

Description

Returns/sets caption alignment.

Syntax

object.**CaptionAlignment** [= *alignment*]

The syntax of the **CaptionAlignment** property has these parts:

Part	Description
object	A List/X+ control.
alignment	An integer specifying the alignment of the caption.

Remarks

This property supports the following values:

Constant	Value	Description
AlignLeft	0	Left Justify
AlignRight	1	Right Justify
AlignCenter	2	Center

Data Type

Integer

CaptionBorderEffect Property

See Also

Description

Returns/sets the caption border effect.

Syntax

object.**CaptionBorderEffect** [= *effect*]

The syntax of the **CaptionBorderEffect** property has these parts:

Part	Description
object	A List/X+ control.
effect	An integer that specifies the style border to use.

Remarks

This property supports the following values:

Constant	Value	Description
BorderEffectNone	0	None
BorderEffectSingle	1	Single line
BorderEffectRaised	2	Raised
BorderEffectPop	3	Pop
BorderEffectDrop	4	Drop
BorderEffectShadow	5	Shadow
BorderEffectInset	6	Inset

Data Type

Integer

IMCaptionPaint Interface CaptionHeight Event

See Also

Description

IMCaptionPaint method called by the control to obtain the caption's height.

Syntax

Sub *object*_**CaptionHeight(**[*index* **As Integer**]**)**

The syntax of the **CaptionHeight** event has these parts:

Part	**Description**
object	Required. An instance of an IMCaptionPaint interface.

Remarks

If you implement the IMCaptionPaint interface in a Class module or UserControl and assign an instance of same to List/X+'s CaptionPaintObject, then the object's CaptionHeight Event will be called as required by List/X+.   Calls to CaptionHeight may occur repeatedly at any time after the object is assigned to the CaptionPaintObject property.

CaptionPaintObject Property

See Also

Description

User draw IMCaptionPaint object.

Syntax

object.**CaptionPaintObject** [= *object*]

The syntax of the **CaptionPaintObject** property has these parts:

Part	Description
object	A List/X+ control.
object	An object that implements the MPaint interface.

Remarks

You can have complete control over what gets displayed in List/X+'s caption by assigning an object that implements the IMPaint interface to the CaptionPaintObject property.

Data Type

Object

Clear Method

See Also

Description

Removes all items from the listbox.

Syntax

object.**Clear**

The syntax of the **Clear** method has these parts:

Part	Description
object	Required. A List/X+ control.

Remarks

Removes all items from the listbox.

MPicture Object Cls Method

See Also

Description

Clears the MPicture object.

Syntax

object.**Cls**

The syntax of the **Cls** method has these parts:

Part	Description
object	Required. An MPicture object.

Remarks

Clears the MPicture object's picture, filling it with the background color.

ColDelimiter Property

See Also

Description

Returns/sets the single character column separator.

Syntax

object.**ColDelimiter** [= *string*]

The syntax of the **ColDelimiter** property has these parts:

Part	Description
object	A List/X+ control.
string	A string expression that specifies the column delimiter character.

Remarks

A single-chracter string used to separate column contents. For instance if ColDelimiter is "," then "foo,bar,baz" specifies the text for three columns.

Data Type

String

ColRowOrder Property

See Also

Description

When True, specifies that arrays passed to PutItems are ordered (column,row). When False, arrays are ordered (row,column). Defaults to True.

Syntax

object.**ColRowOrder** [= *boolean*]

The syntax of the **ColRowOrder** property has these parts:

Part	Description
object	A List/X+ control.
boolean	A boolean expression that determines the order of arrays. When True, arrays passed to PutItems are ordered (column,row). Otherwise, arrays are ordered (row,column).

Remarks

Some objects that return arrays of data, such as RDO resultsets, order the data in (column,row) order instead of (row,column). The ColRowOrder property allows you to specify whether an array that is passed to PutItems will be treated as (column,row) or (row,column).

Data Type

Boolean

Column Object ColumnAlignment Property

Description

Returns/sets column alignment.

Syntax

object.**ColumnAlignment** [= *alignment*]

The syntax of the **ColumnAlignment** property has these parts:

Part	**Description**
object	Required. A Column object.
alignment	An integer that specifies the column alignment.

Remarks

This property supports the following values:

Constant	**Value**	**Description**
AlignLeft	0	Left Justify
AlignRight	1	Right Justify
AlignCenter	2	Center

Data Type

Integer

ColumnIndex Property

See Also

Description

Returns index of the last column clicked on.

Syntax

object.**ColumnIndex**

The syntax of the **ColumnIndex** property has these parts:

Part	**Description**
object	A List/X+ control.

Remarks

This property is set just prior to firing mouse events and contains the index of the column the mouse is over.

Data Type

Integer

Column Object

Properties

Description

Standard VB-style collection, holds a List/X+'s columns, can be used with For Each.

Remarks

Each column has an associated column object, accessed by way of the control's ColumnsCollection object.   Using column objects, you can customize column colors, witdths, alignments, and more.

Columns Property

See Also

Description

Columns Collection.

Syntax

object.**Columns**

The syntax of the **Columns** property has these parts:

Part	**Description**
object	A List/X+ control.

Remarks

A collection containing List/X+'s current columns.   Columns may be added and deleted at runtime.

Data Type

Variant

ColumnsCollection Object

Properties Methods

Description

Object stored in the ColumnsCollection. Provides access to column-specific properties and methods.

Remarks

List/X+ has a collection of columns, accessed via the Columns property. You can add, delete, and invoke column properties and methods using the ColumnsCollection.

IMSort Interface CompareColumns Event

Description

Called by List/X's sorting routine when a column's SortObject is not null.

Syntax

Sub *object*_**CompareColumns(**[*index* **As Integer**,] *str1* **As String**, *str2* **As String**, *column* **As Integer**, **ByRef** *result* **As Integer)**

The syntax of the **CompareColumns** event has these parts:

Part	Description
object	Required. An instance of an IMSort interface.
str1	A string expression to compare.
str2	A string expression to compare.
column	An integer that specifies the column being sorted.
result	Return value. Set this to -1 if str1 < str2; to 0 if str1 = str2; to 1 if str1 > str2.

Remarks

CompareColumns may be called at any time after you assign an object exposing IMSort to a column. When CompareColumns is called, you must compare the passed in column contents and return a result as described in the Syntax description for *result*. The specific algorithm you use to compare column contents is entirely up to you, but if you expect the sort to complete, be sure to be consistent in your comparisons. Returning random, or inconsistent, values during sorting can cause the sort algorithm to run forever.

ColumnsCollection Object Count Property

See Also

Description

Returns the number of columns.

Syntax

object.**Count**

The syntax of the **Count** property has these parts:

Part	Description
object	Required. A ColumnsCollection object.

Remarks

Returns the number of columns in the listbox's columns collection.   The property is read-only.
Columns are added using the collection's Add method and deleted using the Remove method.

Data Type

Integer

MPicture Object CurrentX Property

See Also

Description

Returns/sets the MPicture object's current horizontal location.

Syntax

object.**CurrentX** [= *x*]

The syntax of the **CurrentX** property has these parts:

Part	**Description**
object	Required. An MPicture object.
x	An integer that specifies the current horizontal position.

Remarks

Returns/sets the current x coordinate for the graphics point. The graphics point is specified by CurrentyX and CurrentY and is used as the default parameter for some methods, such as the Line method.

Data Type

Single

MPicture Object CurrentY Property

See Also

Description

Returns/sets the MPicture object's current vertical location.

Syntax

object.**CurrentY** [= *y*]

The syntax of the **CurrentY** property has these parts:

Part	Description
object	Required. An MPicture object.
y	An integer that specifies the current vertical position.

Remarks

Returns/sets the current y coordinate for the graphics point. The graphics point is specified by CurrentX and CurrentY and is used as the default parameter for some methods, such as the Line method.

Data Type

Single

MPicture Object DrawStyle Property

See Also

Description

Returns/sets the MPicture object's drawing style.

Syntax

object.**DrawStyle** [= *style*]

The syntax of the **DrawStyle** property has these parts:

Part	Description
object	Required. An MPicture object.
style	An integer that specifies the current drawing style.

Remarks

This property supports the following values:

Constant	Value	Description
Solid	0	Solid
Dash	1	Dash
Dot	2	Dot
DashDot	3	Dash-Dot
DashDotDot	4	Dash-Dot-Dot
Transparent	5	Transparent
InsideSolid	6	Inside Solid

If DrawWidth is set to a value greater than 1, DrawStyle settings 1 through 4 produce a solid line (the DrawStyle property value isn't changed). If DrawWidth is set to 1, DrawStyle produces the effect described in the preceding table for each setting.

Data Type

Integer

MPicture Object DrawText Method

Description

Paints text on the MPicture object.

Syntax

object.**DrawText***text*,*x1*,*y1*,*x2*,*y2*,*flags*

The syntax of the **DrawText** method has these parts:

Part	Description
object	Required. An MPicture object.
text	Required. A string expression that specifies the text the draw.
x1	Required. A single precision number that specifies the left horizontal coordinate of bounding box for text.
y1	Required. A single precision number that specifies the top vertical coordinate of bounding box for text.
x2	Required. A single precision number that specifies the right horizontal coordinate of bounding box for text.
y2	Required. A single precision number that specifies the bottom vertical coordinate of bounding box for text.
flags	Required. An integer that specifies how to draw the text.

Remarks

Settings for Flags:

Constant	Value	Description
tfTop	0	Aligns text to the top in the bounding box.
tfLeft	0	Aligns text to the left in the bounding box.
tfCenter	1	Centers text horizontally in the bounding box.
tfRight	2	Aligns text to the right in the bounding box.
tfVertCenter	4	Centers text vertically in the bounding box.
tfBottom	8	Aligns text with the bottom of the bounding box.
tfWordbreak	16	Breaks words. Lines are automatically broken between words if a word would extend past the edge of the bounding box. A carriage return-linefeed sequence also breaks the line.
tfSingleLine	32	Displays text on a single line only. Carriage returns and linefeeds do not break the line.
tfNoPrefix	&H800	Turns off processing of prefix characters. Normally, DrawText interprets the mnemonic-prefix character "&" as a directive to underscore the character that follows, and the mnemonic-prefix characters "&&" as a directive to print a single "&". By specifying tfNoPrefix, this processing is turned off.
tfPathEllipsis	&h4000	
tfEndEllipsis	&h8000	Replaces part of the given string with ellipses, if necessary, so that the result fits in the specified bounding box. You can specify tfEndEllipses to replace characters at the end of the string, or dtPathEllipses to replace characters in the middle of the string. If the string contains backslash (\) characters, dtPathEllipses preserves as much as possible of the text after the last backslash.
tfRtlReading	&H20000	Layout in right to left reading order for bi-directional text when the font selected into the hdc is a Hebrew or Arabic font. The default

reading order for all text is left to right.

MPicture Object DrawWidth Property

See Also

Description

Returns/sets the MPicture object's pen width.

Syntax

object.**DrawWidth** [= *long*]

The syntax of the **DrawWidth** property has these parts:

Part	Description
object	Required. An MPicture object.
long	A long integer that specifies the width of the pen, when drawing.

Remarks

Increase the value of this property to increase the width of the line. If the DrawWidth property setting is greater than 1, DrawStyle property settings 1 through 4 produce a solid line (the DrawStyle property value isn't changed). Setting DrawWidth to 1 allows DrawStyle to produce the results shown in the DrawStyle property table.

Data Type

Long

MPicture Object FillColor Property

See Also

Description

Returns/sets the MPicture object's FillColor.

Syntax

object.**FillColor** [= *color*]

The syntax of the **FillColor** property has these parts:

Part	**Description**
object	Required. An MPicture object.
color	A color expression that specifies the fill color to use when drawing.

Remarks

When the FillStyle property is set to its default, 1 (Transparent), the FillColor setting is ignored.

Data Type

Color

MPicture Object FillStyle Property

See Also

Description

Returns or sets the pattern used to fill shapes created with graphics methods.

Syntax

object.**FillStyle** [= *style*]

The syntax of the **FillStyle** property has these parts:

Part	Description
object	Required. An MPicture object.
style	An integer that specifies the fill style.

Remarks

This property supports the following values:

Constant	Value	Description
fsSolid	0	Solid
fsTransparent	1	Transparent
fsHorizontalLine	2	Horizontal Line
fsVerticalLine	3	Vertical Line
fsUpwardDiagonal	4	Upward Diagonal
fsDownwardDiagonal	5	Downward Diagonal
fsCross	6	Cross
fsDiagonalCross	7	Diagonal Cross

When FillStyle is set to 1 (Transparent), the FillColor property is ignored.

Data Type

Integer

Font Property

Description

Returns/sets the listbox's font.

Syntax

*object***.Font** [= *font*]

The syntax of the **Font** property has these parts:

Part	**Description**
object	A List/X+ control.
font	A font specifying the font to use.

Data Type

Font

MPicture Object Font Property

See Also

Description

Returns/sets the MPicture object's Font.

Syntax

object.**Font** [= *font*]

The syntax of the **Font** property has these parts:

Part	Description
object	Required. An MPicture object.
font	A font expression that specifies the font to use when drawing text.

Remarks

Specifies the font used by the DrawText method.

Data Type

Font Object

MPicture Object FontTransparent Property

Description

Returns or sets a value that determines whether or not the background is displayed in the spaces around characters.

Syntax

object.**FontTransparent** [= *boolean*]

The syntax of the **FontTransparent** property has these parts:

Part	Description
object	Required. An MPicture object.
boolean	A boolean expression that determines if the font's background is transparent.

Remarks

Setting this property to True allows you to paint text over graphics without overwriting the graphics and text behind the new text.

.

Column Object ForeColor Property

See Also

Description

Returns/sets the column's foreground (text) color.

Syntax

object.**ForeColor** [= *color*]

The syntax of the **ForeColor** property has these parts:

Part	Description
object	Required. A Column object.
color	A color expression that specifies the text color of this column.

Remarks

Specifies the foreground (text) color for a column. The listbox itself and individual list items also have foreground color properties. When a column for some item is about to be displayed, the item's ForeColor property is checked. If it has been set, then that color is used. If the item's ForeColor has not been set, then the column's ForeColor is used (if set). Otherwise, the listbox's ForeColor is used.

Data Type

Color

MPicture Object ForeColor Property

See Also

Description

Returns/sets the color used to draw text or graphics.

Syntax

object.**ForeColor** [= *color*]

The syntax of the **ForeColor** property has these parts:

Part	Description
object	Required. An MPicture object.
color	A color expression that specifies the text color.

Remarks

The color used when drawing text or graphics.

Data Type

Color

ForeColor Property

See Also

Description

Returns/sets the listbox's foreground color.

Description

Returns or sets the color of the control's text.

Syntax

object.**ForeColor** [= *color*]

The syntax of the **ForeColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression specifying the color to use for the text.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different ForeColor properties.

Each list item has a ForeColor property.   If the ForeColor for a given item has been specified then that color is used as the ForeColor for each of that item's columns.

Also each Column has a ForeColor property.   If a value other than -1 is assigned to a Column's ForeColor property and the Item's ForeColor property is -1, then the color specified by the Column's ForeColor property is used.

If neither of the Column or Item ForeColors has been specified then the listbox's ForeColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

MPicture Object hDC Property

Description

Returns the MPicture object's hDC.

Syntax

object.**hDC**

The syntax of the **hDC** property has these parts:

Part	Description
object	Required. An MPicture object.

Remarks

This property is read-only.   The hDC property provides access to the MPicture object's hDC for advanced drawing operations using the Windows API.

Data Type

Long

Column Object HeaderDisplayable Property

See Also

Description

Returns True if a column's header is visible.

Syntax

object.**HeaderDisplayable**

The syntax of the **HeaderDisplayable** property has these parts:

Part	Description
object	Required. A Column object.

Remarks

Column headers will be displayed if at least one column heading has text, at least one column heading is Visible, and at least one column heading's width is greater than zero

Data Type

Boolean

Column Object Heading Property

See Also

Description

Text displayed in column heading.

Syntax

object.**Heading** [= *string*]

The syntax of the **Heading** property has these parts:

Part	Description
object	Required. A Column object.
string	A string that determines the text displayed in a column heading.

Remarks

If any column object has a non-zero Width, has the Visible property set to True, and has a non-null string for the Heading property, then column headings are displayed for all columns.

The string in Heading is displayed in the column heading. Multiple-line headings are created by using CR inside the Heading string. For instance, the following code creates a two-line header for column zero.

```
    MList1.Columns(0).Heading = "Column" & Chr(13) & "Heading"
```

The height of all headings is adjusted to match the height of the heading having the greatest number of lines

Data Type

String

Column Object HeadingAlignment Property

See Also

Description

Returns or sets the column's heading alignment.

Syntax

object.**HeadingAlignment** [= *align*]

The syntax of the **HeadingAlignment** property has these parts:

Part	Description
object	Required. A Column object.
align	An integer that specifies the heading alignment.

Remarks

This property supports the following values:

Constant	Value	Description
AlignLeft	0	Left Justify
AlignRight	1	Right Justify
AlignCenter	2	Center

Data Type

Integer

HeadingsBorderEffect Property

See Also

Description

Returns/sets the listbox's headings' border effect

Syntax

object.**HeadingsBorderEffect** [= *effect*]

The syntax of the **HeadingsBorderEffect** property has these parts:

Part	Description
object	A List/X+ control.
effect	An integer that specifies the style border to use.

Remarks

This property supports the following values:

Constant	Value	Description
BorderEffectNone	0	None
BorderEffectSingle	1	Single line
BorderEffectRaised	2	Raised
BorderEffectPop	3	Pop
BorderEffectDrop	4	Drop
BorderEffectShadow	5	Shadow
BorderEffectInset	6	Inset

Data Type

Integer

Column Object HiliteColor Property

See Also

Description

Returns/sets the column's highlighted background color.

Syntax

object.**HiliteColor** [= *color*]

The syntax of the **HiliteColor** property has these parts:

Part	Description
object	Required. A Column object.
color	A color expression that specifies the background color of an item, when highlighted.

Remarks

Specifies the background color for a column in a selected item. The listbox itself and individual list items also have HiliteColor properties. When a column for some item is about to be displayed and the item is selected, the item's HiliteColor property is checked. If it has been set then that color is used. If the item's HiliteColor has not been set then the column's HiliteColor is used if it has been set, otherwise the listbox's HiliteColor is used.

Setting the HiliteColor property to -1 causes the HiliteColor property to be treated as if it were never set.

Data Type

Color

HiliteColor Property

See Also

Description

Background color for selected items.

Syntax

object.**HiliteColor** [= *color*]

The syntax of the **HiliteColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression that specifies the background color of the item, when highlighted.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different backcolor properties.

Each list item has HiliteTextColor and HiliteColor properties.   If the HiliteColor for a given item has been specified then that color is used as the HiliteColor for each of that item's columns.

Also each Column has a HiliteColor property.   If a value other than -1 is assigned to a Column's HiliteColor property and the Item's HiliteColor property is -1 then the color specified by the Column's HiliteColor property is used.

If neither of the Column or Item HiliteColors has been specified then the listbox's HiliteColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

Column Object HiliteTextColor Property

See Also

Description

Returns/sets the column's highlighted foreground color.

Syntax

object.**HiliteTextColor** [= *color*]

The syntax of the **HiliteTextColor** property has these parts:

Part	Description
object	Required. A Column object.
color	A color expression that specifies the text color of an item, when highlighted.

Remarks

Specifies the foreground (text) color for a column in a selected item. The listbox itself and individual list items also have HiliteTextColor properties.   When a column for some item is about to be displayed and the item is selected, the item's HiliteTextColor property is checked.   If it has been set then that color is used.   If the item's HiliteTextColor has not been set then the column's HiliteTextColor is used if it has been set, otherwise the listbox's HiliteTextColor is used.

Setting the HiliteTextColor property to -1 causes the HiliteTextColor property to be treated as if it were never set.

Data Type

Color

HiliteTextColor Property

See Also

Description

Text color for selected items.

Syntax

object.**HiliteTextColor** [= *color*]

The syntax of the **HiliteTextColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression that specifies the text color of the item, when highlighted.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different forecolor properties.

Each list item has HiliteTextColor and HiliteColor properties. If the HiliteColor for a given item has been specified then that color is used as the HiliteColor for each of that item's columns.

Also each Column has a HiliteTextColor property.   If a value other than -1 is assigned to a Column's HiliteTextColor property and the Item's HiliteTextColor property is -1 then the color specified by the Column's HiliteTextColor property is used.

If neither of the Column or Item HiliteTextColors has been specified then the listbox's HiliteTextColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

HorzDivForeColor Property

See Also

Description

Returns/sets the horizontal divider foreground color.

Syntax

object.**HorzDivForeColor** [= *color*]

The syntax of the **HorzDivForeColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression that specifies the color of the horizontal dividers.

Remarks

This color is used to draw the horizontal dividers, if any.

Data Type

Color

HorzDividers Property

Description

Controls the display of horizontal lines between rows.

Syntax

object.**HorzDividers** [= *effects*]

The syntax of the **HorzDividers** property has these parts:

Part	Description
object	A List/X+ control.
effects	An integer that specifies the style of the horizontal dividers.

Remarks

This property supports the following values:

Constant	Value	Description
DividerEffectNone	0	None
DividerEffectSingle	1	Single line
DividerEffectDash	2	Dash
DividerEffectDot	3	Dot
DividerEffectDashDot	4	Dash-Dot
DividerEffectDashDotDot	5	Dash-Dot-Dot

Data Type

Integer

ColumnsCollection Object Item Property

See Also

Description

Returns a column object.

Syntax

object.**Item(** *id* **)**

The syntax of the **Item** property has these parts:

Part	Description
object	Required. A ColumnsCollection object.
id	An integer specifying the column object desired.

Remarks

Returns the column object specified by the integer argument. The first column has an index of zero, the highest numbered column is given by the Count property.   If Count is zero there are no columns.

Data Type

Variant

ItemBackColor Property

See Also

Description

Returns/sets the background color for a single item.

Syntax

object.**ItemBackColor(** *index* **)** [= *color*]

The syntax of the **ItemBackColor** property has these parts:

Part	**Description**
object	A List/X+ control.
index	A long integer specifying an item.
color	A color expression that specifies the background color of the item.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different backcolor properties.

Each list item has ForeColor and BackColor properties. If the ForeColor for a given item has been specified then that color is used as the ForeColor for each of that item's columns.

Also each Column has a ForeColor property.   If a value other than -1 is assigned to a Column's ForeColor property and the Item's ForeColor property is -1, then the color specified by the Column's HiliteTextColor property is used.

If neither of the Column or Item ForeColor has been specified then the listbox's ForeColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

ItemData Property

See Also

Description

Returns/sets an optional Variant value for each item.

Syntax

*object***.ItemData** [= *index*, *variant*]

The syntax of the **ItemData** property has these parts:

Part	**Description**
object	A List/X+ control.
index	A long integer specifying an item.
variant	A variant expression that specifies data to associate with the list item.

Remarks

You may assign a Variant as the ItemData for each item in the list. Since List/X+'s ItemData is a Variant you can associate anything that can be stored in a Variant with a list item, including arbitrarily complex objects.

Data Type

Variant

ItemForeColor Property

See Also

Description

Returns/sets the foreground (text) color.

Syntax

object.**ItemForeColor(** *index* **)** [= *color*]

The syntax of the **ItemForeColor** property has these parts:

Part	**Description**
object	A List/X+ control.
index	A long integer specifying an item.
color	A color expression that specifies the text color of the item.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different forecolor properties.

Each list item has ItemForeColor and ItemBackColor properties. If the ItemForeColor for a given item has been specified then that color is used as the ItemForeColor for each of that item's columns.

Also each Column has a ForeColor property.   If a value other than -1 is assigned to a Column's ForeColor property and the Item's ForeColor property is -1, then the color specified by the Column's HiliteTextColor property is used.

If neither of the Column or ItemForeColor has been specified then the listbox's ForeColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

IMListPaint Interface ItemHeight Event

Description

Called by the control to obtain the height for a given item.

Syntax

Sub *object*_**ItemHeight(**[*index* **As Integer**,] *item* **As Long**, **ByRef** *itemheight* **As Integer**, **ByRef** *heightvalid* **As Boolean)**

The syntax of the **ItemHeight** event has these parts:

Part	Description
object	Required. An instance of an IMListPaint interface.
item	A long integer that specifies the item to measure.
itemheight	An integer that the method sets to specify the height of the item.
heightvalid	A boolean expression that the method sets to specify whether or not the value in *itemheight* is valid (output).

Remarks

This method may be called at any time by the control to measure the height of a given Item. The height is returned to the control via the ItemHeight parameter.

If you are displaying multiple columns, be sure to set ItemHeight to the height of the tallest column for the specified Item.

ItemHiliteTextColor Property

See Also

Description

Returns/sets the highlight foreround color for a single item.

Syntax

object.**ItemHiliteTextColor** [= *index*, *color*]

The syntax of the **ItemHiliteTextColor** property has these parts:

Part	Description
object	A List/X+ control.
index	A long integer specifying an item.
color	A color expression that specifies the text color of the item, when highlighted.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different forecolor properties.

Each list item has HiliteTextColor and HiliteColor properties. If the HiliteColor for a given item has been specified then that color is used as the HiliteColor for each of that item's columns.

Also each Column has a HiliteColor property.   If a value other than -1 is assigned to a Column's HiliteColor property and the Item's HiliteColor property is -1 then the color specified by the Column's HiliteColor property is used.

If neither of the Column or Item HiliteColor has been specified then the listbox's HiliteColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

ItemHiliteColor Property

See Also

Description

Returns/sets the highlight background color for a single item.

Syntax

object.**ItemHiliteColor(** *index* **)** [= *color*]

The syntax of the **ItemHiliteColor** property has these parts:

Part	Description
object	A List/X+ control.
index	A long integer specifying an item.
color	A color expression that specifies the background color of the item, when highlighted.

Remarks

The colors used when displaying a particular column in a given row are controlled by three different backcolor properties.

Each list item has HiliteTextColor and HiliteColor properties. If the HiliteColor for a given item has been specified then that color is used as the HiliteColor for each of that item's columns.

Also each Column has a HiliteColor property.   If a value other than -1 is assigned to a Column's HiliteColor property and the Item's HiliteColor property is -1 then the color specified by the Column's HiliteColor property is used.

If neither of the Column or Item HiliteColor has been specified then the listbox's HiliteColor property is used when displaying a given row/column.

Any number of color combinations can be used if you draw items yourself by assigning an appropriate object to columns' PaintObject properties.

Data Type

Color

MPicture Object Line Method

Description

Draws lines and boxes on the MPicture object.

Syntax

*object***.Line Step (***x1*,*y1***) - Step (***x2*,*y2***)**,*color,***BF**

The syntax of the **Line** method has these parts:

Part	**Description**
object	Required. An MPicture object.
Step	Optional. A keyword that specifies that the coordinates given are relative to the current graphics position (CurrentX and CurrentY).
x1	Optional. A single value indicating the horizontal coordinate of the starting point for the line or rectangle. The ScaleMode property determines the unit of measure used. If omitted, the line begins at the position indicated by CurrentX.
y1	Optional. A single value indicating the vertical coordinate of the starting point for the line or rectangle. The ScaleMode property determines the unit of measure used. If omitted, the line begins at the position indicated by CurrentY.
Step	Optional. A keyword that specifies that the coordinates given are relative to the starting point.
x2	Optional. A single value indicating the horizontal coordinate of the end point for the line or rectangle.
y2	Optional. A single value indicating the vertical coordinate of the end point for the line or rectangle.
color	Optional. A long integer value indicating the RGB color used to draw the line. If omitted, the ForeColor property setting is used.
B	Optional. A keyword that specifies that the method should draw a box, rather than a line. The coordinates specified determine opposing corners of the box.
F	Optional. A keyword that, when used with the B keyword, specifies that the box should be filled with the current ForeColor.

Remarks

To draw connected lines, begin a subsequent line at the end point of the previous line.

The width of the line drawn depends on the setting of the DrawWidth property. The way a line or box is drawn on the background depends on the setting of the DrawMode and DrawStyle properties.

When Line executes, the CurrentX and CurrentY properties are set to the end point specified by the arguments.

This method cannot be used in a With/End With block.

List Property

See Also

Description

Returns/sets a listbox item.

Syntax

object.**List(** *index* **)** [= *string*]

The syntax of the **List** property has these parts:

Part	**Description**
object	A List/X+ control.
index	A long integer specifying the position for the item.
string	A string expression specifying the replacement string.

Remarks

Returns/sets list items. *index* must be between zero and ListCount-1.

Data Type

String

ListBorderEffect Property

See Also

Description

Returns/sets the listbox's border effect.

Syntax

object.**ListBorderEffect** [= *effect*]

The syntax of the **ListBorderEffect** property has these parts:

Part	Description
object	A List/X+ control.
effect	An integer that specifies the style border to use.

Remarks

This property supports the following values:

Constant	Value	Description
BorderEffectNone	0	None
BorderEffectSingle	1	Single line
BorderEffectRaised	2	Raised
BorderEffectPop	3	Pop
BorderEffectDrop	4	Drop
BorderEffectShadow	5	Shadow
BorderEffectInset	6	Inset

Data Type

Integer

ListCount Property

Description

Returns the number of items in the listbox.

Syntax

*object***.ListCount**

The syntax of the **ListCount** property has these parts:

Part	**Description**
object	A List/X+ control.

Data Type

Long

ListIndex Property

Description

Returns or sets the index of the currently selected item in the control.

Syntax

object.**ListIndex** [= *index*]

The syntax of the **ListIndex** property has these parts:

Part	**Description**
object	A List/X+ control.
index	A long integer specifying the currently selected item.

Remarks

For a control in which users can make multiple selections, this property's behavior depends on the number of items selected. If only one item is selected, ListIndex returns the index of that item. In a multiple selection, ListIndex returns the index of the item contained within the focus rectangle, whether or not that item is actually selected.

Data Type

Long

Column Object MaxWidth Property

See Also

Description

Returns/sets the maximum width to which a user can resize a column.

Syntax

object.**MaxWidth** [= *long*]

The syntax of the **MaxWidth** property has these parts:

Part	Description
object	Required. A Column object.
long	A long integer that specifies the maximum column width.

Remarks

If the UserResizeEnabled property is True, the user can resize the columns. The maximum width to which a user can size a column is specified by the MaxWidth property. If MaxWidth is -1, there is no bound on the maximum width.

Data Type

Integer

IMCaptionPaint Interface

Events

Description

Interface implemented by the programmer when creating a COM object that can be assigned to the list's CaptionPaintObject property.

Remarks

List/X+ provides a CaptionPaintObject property which provides you with complete control over the appearance of the list's caption.   If you assign some object to a column's PaintObject, List/X+ will attempt to use that object to paint the caption as required.

In order for List/X+ to successfully use an object assigned to the CaptionPaintObject property, the object must either Implement IMCaptionPaint and its methods, or the paint object must expose two publicly available methods named IMCaptionPaint_CaptionHeight and IMCaptionPaint_PaintCaption.

When programming with VB5, or later versions of Visual Basic, Implements is the preferred approach since it causes VB to use early-binding with its performance enhancements.

When using List/X+ with VB4, you have no choice but to simply expose the IMCaptionPaint_ItemHeight and IMCaptionPaint_PaintCaption methods in a class module.   The number and types of arguments are identical to those used with Implements.

For instance, when programming in VB5 using Implements, one of the functions you implement is:

```
    Private Function IMCaptionPaint_CaptionHeight(height As Integer) As
Boolean
```

In VB4 you would declare the following function:

```
    Public Function IMCaptionPaint_CaptionHeight(height As Integer) As
Boolean
```

Note that the function must be public since List/X+ must use OLE Invoke to access it.

Column Object MinWidth Property

See Also

Description

Returns or sets the minimum width to which a column can be resized.

Syntax

object.**MinWidth** [= *width*]

The syntax of the **MinWidth** property has these parts:

Part	Description
object	Required. A Column object.
width	A long integer that specifies the minimum column width.

Remarks

If the UserResizeEnabled property is True, the user can resize the columns.   The minimum width to which a user can size a column is specified by the MinWidth property. If MinWidth is zero, there is no lower bound on a column's width.

Data Type

Integer

IMListPaint Interface

Events

Description

Interface implemented by the programmer when creating a COM object that can be assigned to a column's PaintObject property.

Remarks

Each of List/X+'s columns has a PaintObject property.   If you assign some object to a column's PaintObject, List/X+ will attempt to use that object to paint the column, or the column's header, as required.   Using PaintObjects, you have complete control over the appearance of a column.

In order for List/X+ to successfully use an object assigned to a column's PaintObject property, the object must either Implement IMListPaint and its methods, or the paint object must expose two publicly available methods named IMListPaint_ItemHeight and IMListPaint_PaintColumn.

When programming with VB5, or later versions of Visual Basic, Implements is the preferred approach since it causes VB to use early-binding with its performance enhancements.

When using List/X+ with VB4, you have no choice but to simply expose the IMListPaint_ItemHeight and IMListPaint_PaintColumn as public methods in a class module.   The number and types of arguments are identical to those used with Implements.

For instance, when programming in VB5 using Implements, one of the functions you implement is:

```
    Private Function IMListPaint_ItemHeight(ByVal Item As Long, ItemHeight
As Integer) As Boolean
```

In VB4 you would declare the following function:

```
    Public Function IMListPaint_ItemHeight(ByVal Item As Long, ItemHeight As
Integer) As Boolean
```

Note that the function must be public since List/X+ must use OLE Invoke to access it.

MouseOverCaption Property

See Also

Description

Returns True when the mouse is over the caption.

Syntax

object.**MouseOverCaption**

The syntax of the **MouseOverCaption** property has these parts:

Part	Description
object	A List/X+ control.

Remarks

Before each mouse-related event is fired, the MouseOverCaption property is set to True if the mouse is over the heading.   This property is not reliable outside of mouse-related event procedures.

Data Type

Boolean

MouseOverList Property

See Also

Description

Returns True when the mouse is over the list part of the listbox.

Syntax

object.**MouseOverList**

The syntax of the **MouseOverList** property has these parts:

Part	**Description**
object	A List/X+ control.

Remarks

Before each mouse-related event is fired, the MouseOverList property is set to True if the mouse is over the list part of the listbox. This property is not reliable outside of mouse-related event procedures.

Data Type

Boolean

MouseOverHeading Property

See Also

Description

Returns True when the mouse is over a heading.

Syntax

object.**MouseOverHeading**

The syntax of the **MouseOverHeading** property has these parts:

Part	Description
object	A List/X+ control.

Remarks

Before each mouse-related event is fired the MouseOverHeading property is set to True if the mouse is over the heading.   This property is not reliable outside of mouse-related event procedures.

Data Type

Boolean

MousePointer Property

Description

Returns/sets the listbox mouse pointer.

Syntax

object.**MousePointer** [= *mousepointer*]

The syntax of the **MousePointer** property has these parts:

Part	Description
object	A List/X+ control.
mousepointer	An integer specifying the style of the mouse pointer, when over this control.

Remarks

This property supports the following values:

Constant	Value	Description
Default	0	Default
Arrow	1	Arrow
Cross	2	Cross
IBeam	3	I-Beam
Icon	4	Icon
Size	5	Size
SizeNESW	6	Size NE SW
SizeNS	7	Size N S
SizeNWSE	8	Size NW SE
SizeEW	9	Size W E
UpArrow	10	Up Arrow
Hourglass	11	Hourglass
NoDrop	12	No Drop
ArrowHourglass	13	Arrow and Hourglass
ArrowQuestion	14	Arrow and Question
SizeAll	15	Size All

Data Type

Integer

MPicture Object

Properties Methods

Description

VB PictureBox-compatible object passed to IMListPaint_PaintColumn or IMCaptionPaint_Paint caption methods.

Remarks

List/X+ passes an MPicture object to IMListPaint_PaintColumn and IMCaptionPaint_PaintCaption. MPicture objects are compatible with VB's Picture object, providing easy access to the drawing methods with which you're already familiar. Upon return from one of the user draw functions the contents of the MPicture object are displayed.

IMSort Interface

Events

Description

Interface implemented by the programmer when creating a COM object that can be assigned to a column's SortObject property.

Remarks

Each of List/X+'s columns has a SortObject property.   If you assign an object that implements the IMSort interface to a column's SortObject property, then the object's CompareColumns method will be callec by List/X+ whenever the list is being sorted.

When programming with VB5, or later versions of Visual Basic, Implements is the preferred approach since it causes VB to use early-binding with its performance enhancements.

When using List/X+ with VB4, you have no choice but to simply expose the IMSort_CompareColumns as a public method in a class module.   The number and types of arguments are identical to those used with Implements.

For instance, when programming in VB5 using Implements, implement:

```
    Private Function IMSort_CompareColumns(item1 As String, item2 As String,
col As Integer) As Integer
```

In VB4 you would declare the following function:

```
    Public Function IMSort_CompareColumns(item1 As String, item2 As String,
col As Integer) As Integer
```

Note that the function must be public since List/X+ must use OLE Invoke to access it.

MultiSelect Property

Description

Returns or sets a value that determines whether or not a user can make multiple selections and how they can be made.

Syntax

object.**MultiSelect** [= *multi*]

The syntax of the **MultiSelect** property has these parts:

Part	Description
object	A List/X+ control.
multi	An integer specifying the style of multiselect to use.

Remarks

This property supports the following values:

Constant	Value	Description
MultiSelectNone	0	None
MultiSelectSimple	1	Simple
MultiSelectExtended	2	Extended

Data Type

Integer

IMCaptionPaint Interface PaintCaption Event

See Also

Description

IMCaptionPaint method called by the control when the caption needs to be painted.

Syntax

Sub *object*_**PaintCaption(**[*index* **As Integer**]**)**

The syntax of the **PaintCaption** event has these parts:

Part	**Description**
object	Required. An instance of an IMCaptionPaint interface.

Remarks

If you implement the IMCaptionPaint interface in a Class module or UserControl and assign an instance of same to List/X+'s CaptionPaintObject, then the object's PaintCaption Event will be called as required by List/X+.   Calls to PaintCaption may occur repeatedly at any time after the object is assigned to the CaptionPaintObject property.

IMListPaint Interface PaintColumn Event

Description

Called by the control when a particular row/column needs to be painted.

Syntax

Sub *object*_**PaintColumn(**[*index* **As Integer**,] **ByRef** *picture* **As IMPicture**, *item* **As Long**, *column* **As Long**, *selected* **As Boolean**, **ByRef** *painted* **As Boolean)**

The syntax of the **PaintColumn** event has these parts:

Part	Description
object	Required. An instance of an IMListPaint interface.
picture	An IMPaint object.
item	A long integer that specifies the item to paint.
column	A long integer that specifies the column to paint.
selected	A boolean expression that specifies whether or not the item is selected.
painted	A boolean expression that the method sets to specify whether or not it painted the item (output).

Remarks

The List/X+ control calls the Paint method before painting a column if the column has an object having an IMListPaint interface assigned to it.

When the Paint method is called, the Painted parameter is set to True. If you do not paint the column, Painted must be set to False when returning from the function in order for default painting to occur.

Column Object PaintObject Property

See Also

Description

Returns/sets an object that implements the MPaint interface.

Syntax

object.**PaintObject** [= *paintobject*]

The syntax of the **PaintObject** property has these parts:

Part	Description
object	Required. A Column object.
paintobject	An object implementing the IMListPaint interface.

Remarks

A column's PaintObject property provides the ultimate control over how a column is displayed. If you assign an object that implements IMListPaint, then List/X+ will invoke methods belonging to that PaintObject. One of the methods is called to obtain the height of a particular column in a particular list item. The other method provides a VB compatible Paint object on which you can draw. The results are displayed in the column.

Data Type

MPaint Object

MPicture Object PaintPicture Method

Description

Paints a picture onto the picture object.

Syntax

*object***.PaintPicture***picture*,*x1*,*y1*,*width1*,*height1*,*x2*,*y2*,*width2*,*height2*,*opcode*

The syntax of the **PaintPicture** method has these parts:

Part	Description
object	Required. An MPicture object.
picture	Required. The source of the graphic to be drawn onto object. Must be the Picture property of a Form or PictureBox.
x1	Required. A single precision number indicating the destination x-coordinate.
y1	Required. A single precision number indicating the destination y-coordinate.
width1	Required. A single precision number indicating the destination width.
height1	Required. A single precision number indicating the destination height.
x2	Required. A single precision number indicating the clipping x-coordinate.
y2	Required. A single precision number indicating the clipping y-coordinate.
width2	Required. A single precision number indicating the clipping width.
height2	Required. A single precision number indicating the clipping height.
opcode	Required. A long integer that is used only with bitmaps. It defines a bit-wise operation (such as vbMergeCopy or vbSrcAnd) that is performed on picture as it's drawn on object. For a complete list of bit-wise operator constants, see the RasterOp Constants topic in Visual Basic Help.

Remarks

Using this method, you can flip a bitmap by using negative values for the destination height (height1) and/or width (width1), depending on which axis you wish to flip.

You can omit trailing arguments.   If you omit an optional trailing argument, don't use any commas after that last argument you specify.   If you want to specify an optional argument, you must specify all of the optional arguments preceding it.

You can omit as many optional trailing arguments as you want.   If you omit an optional trailing argument or arguments, don't use any commas following the last argument you specify. If you want to specify an optional argument, you must specify all optional arguments that appear in the syntax before it.

MPicture Object Picture Property

See Also

Description

Returns/sets MPicture object.

Syntax

object.**Picture** [= *picture*]

The syntax of the **Picture** property has these parts:

Part	Description
object	Required. An MPicture object.
picture	A picture to be displayed.

Remarks

The MPicture object's Picture property is used to set or retrieve the picture that will be drawn into the column being painted after the control's MPaint_PaintColumn method returns.   You should treat the MPicture object as if it were created just prior to the MPaint_PaintColumn method and destroyed immediately after.   Nothing about the object is guaranteed to persist between occurences of the MPaint_PaintColumn method.

The MPicture object is identical in every respect to the VB Picture object and you can do with it anything you can do with VB's Picture object.

After the MPaint_PaintColumn method returns, the MPicture object is displayed in the rectangle defined by the Item and Column being painted.

Data Type

Picture

MPicture Object Point Method

See Also

Description

Returns the color value of a point.

Syntax

color **=** *object***.Point(** *x*,*y* **)**

The syntax of the **Point** method has these parts:

Part	**Description**
object	Required. An MPicture object.
color	Optional. A long integer variable that receives the color value of the point specified.
x	Optional. A single precision number indicating the horizontal (x-axis) coordinate.
y	Optional. A single precision number indicating the vertical (y-axis) coordinate.

Remarks

If the point referred to by the x and y coordinates is outside the object, the Point method returns -1.

Data Type

Long

MPicture Object PSet Method

See Also

Description

Returns a point's color.

Syntax

*object***.PSet Step (***x*,*y***)**,*color*

The syntax of the **PSet** method has these parts:

Part	Description
object	Required. An MPicture object.
Step	Optional. A keyword that specifyies that the coordinates given are relative to the current position given by the CurrentX and CurrentY properties.
x	Optional. A single precision number indicating the horizontal (x-axis) coordinate.
y	Optional. A single precision number indicating the vertical (y-axis) coordinate.
color	Optional. A long integer specifying the color to paint.

Remarks

If *x* or *y* is not specified, then CurrentX or CurrentY (or both) is used.   If color is not specified, then ForeColor is used.

PutItems Method

See Also

Description

Adds an array of strings or variant arrays to the list, beginning at Index.   Existing items are replaced, new items are added as needed.

Syntax

object.**PutItems***Items*,*index*

The syntax of the **PutItems** method has these parts:

Part	**Description**
object	Required. A List/X+ control.
Items	Required. An array of strings or variants.
index	Required. A variant expression that specifies the start index.

Remarks

Use the PutItems method to add an array of values to the list

The Items array can be either a 1-dimensional array of string or variants, or a 2-dimensional array of strings or variants.   The 2-dimensional arrays are used with multicolumn lists.

The ColRowOrder property determines whether 2-dimensional arrays are regarded as being in (column,row) or (row,column) order.

RClick Event

Description

Occurs when the user presses and then releases the right mouse button over an object.

Syntax

Sub *object*_**RClick(**[*index* **As Integer**]**)**

The syntax of the **RClick** event has these parts:

Part	**Description**
object	A List/X+ control.
index	An integer that identifies a control if it's in a control array.

Remarks

The Click event is fired only for the left mouse button.   The RClick event is fired only when the right mouse button is used.

ColumnsCollection Object Remove Method

See Also

Description

Removes one column.

Syntax

object.**Remove***index*

The syntax of the **Remove** method has these parts:

Part	Description
object	Required. A ColumnsCollection object.
index	Optional. An integer specifying the column to remove.

Remarks

Remove one column from the columns collection.   Columns are numbered from zero to Count-1. Count is zero if there are no columns.

RemoveItem Method

See Also

Description

Removes an item from the listbox.

Syntax

object.**RemoveItem***index*

The syntax of the **RemoveItem** method has these parts:

Part	**Description**
object	Required. A List/X+ control.
index	Required. A long integer specifying the item to remove from the list.

Remarks

Removes the specified item from the listbox.   Index must be within the the range zero to ListCount - 1, inclusive.

ReplaceItem Method

See Also

Description

Replaces an item in the listbox.

Syntax

*object***.ReplaceItem***item*,*index*

The syntax of the **ReplaceItem** method has these parts:

Part	**Description**
object	Required. A List/X+ control.
item	Required. A string expression specifying the replacement string.
index	Required. A long integer specifying the position for the item.

Remarks

This method is equivalent to RemoveItem followed by AddItem, except that it's faster and causes fewer screen updates.

MPicture Object ScaleHeight Property

See Also

Description

Returns the MPicture object's height.

Syntax

object.**ScaleHeight**

The syntax of the **ScaleHeight** property has these parts:

Part	Description
object	Required. An MPicture object.

Remarks

Returns the height of the MPicture object expressed in the units specified by the ScaleMode property.

Data Type

Single

MPicture Object ScaleMode Property

See Also

Description

Returns or sets a value that determines the unit of measurement for coordinates of an object.

Syntax

object.**ScaleMode** [= *mode*]

The syntax of the **ScaleMode** property has these parts:

Part	Description
object	Required. An MPicture object.
mode	An integer specifying the unit of measurement.

Remarks

This property supports the following values:

Constant	Value	Description
Twip	1	Twip
Point	2	Point
Pixel	3	Pixel
Character	4	Character
Inches	5	Inches
Millimeter	6	Millimeter
Centimeter	7	Centimeter

Data Type

Integer

MPicture Object ScaleWidth Property

Description

Returns the MPicture object's width.

Syntax

object.**ScaleWidth**

The syntax of the **ScaleWidth** property has these parts:

Part	Description
object	Required. An MPicture object.

Remarks

Returns the width of the MPicture object expressed in the units specified by the ScaleMode property.

Data Type

Single

Sorted Property

See Also

Description

Returns/sets a boolean that determines whether or not the listbox is sorted.

Syntax

object.**Sorted** [= *boolean*]

The syntax of the **Sorted** property has these parts:

Part	**Description**
object	A List/X+ control.
boolean	A boolean expression that specifies whether or not the list is sorted.

Remarks

Determines whether the list items will be displayed in sorted order or in index order.

Data Type

Boolean

Column Object SortObject Property

See Also

Description

Returns/sets an object that implements IMSort interface.

Syntax

object.**SortObject**

The syntax of the **SortObject** property has these parts:

Part	Description
object	Required. A Column object.

Remarks

You may assign a sort object to each column.   If you do so, the sort object's comparison method will be called whenever Lixt/X+ needs to compare the contents   of two columns for sorting.

Using sort objects, you can control exactly how items are sorted in the list.

Data Type

IMSort Object

SortOrder Property

See Also

Description

Returns/sets a string that specifies the sort order for each column.

Syntax

object.**SortOrder** [= *string*]

The syntax of the **SortOrder** property has these parts:

Part	**Description**
object	A List/X+ control.
string	A string that specifies the sort order of the list.

Remarks

Using the SortOrder property, you can sort the listbox in either ascending or descending order on a per column basis.

The string assigned to the SortOrder property must be of the form "<col#><direction> <col#><direction> ...".   For instance, the string "0a 1d 2a" will sort the list such that the first column is sorted ascending, the second descending, and the third ascending.

Note that the order of sorting may be significant, depending upon the data in the list.   "2a 1a 0a" probably will not produce the same list order as "0a 1a 2a".

You may omit the column number, in which case the column is assumed to be either zero or one greater than the previous column.   The following are all valid sort order specifications:

a d a
2a d
a 3a

Data Type

String

TabStop Property

Description

Returns or sets a value which determines whether or not the user can navigate to the control using the Tab key.

Syntax

object.**TabStop** [= *boolean*]

The syntax of the **TabStop** property has these parts:

Part	Description
object	A List/X+ control.
boolean	A boolean expression that determines tabs will bring focus to this control.

Remarks

Normally, this property should be set to True (the default).

Data Type

Boolean

MPicture Object TextHeight Method

See Also

Description

Returns the height a string will be if drawn on the MPicture object.

Syntax

height **=** *object***.TextHeight(** *text* **)**

The syntax of the **TextHeight** method has these parts:

Part	Description
object	Required. An MPicture object.
height	Optional. An integer variable that receives the height of the text.
text	Required. A string expression to measure.

Remarks

Uses the current Font to calculate and return the height of the given string

Data Type

Integer

MPicture Object TextWidth Method

See Also

Description

Returns the width a string will be if drawn on the MPicture object.

Syntax

width **=** *object***.TextWidth(** *text* **)**

The syntax of the **TextWidth** method has these parts:

Part	Description
object	Required. An MPicture object.
width	Optional. An integer variable that receives the width of the text.
text	Required. A string expression to measure.

Remarks

Uses the current Font to calculate and return the width of the given string

Data Type

Integer

ThreedDarkShadowColor Property

See Also

Description

Returns/sets the listbox's 3D dark shadow color.

Syntax

object.**ThreedDarkShadowColor** [= *color*]

The syntax of the **ThreedDarkShadowColor** property has these parts:

Part	Description
object	A List/X+ control.
color	A color expression that specifies the color to use for the dark portion of the shadow.

Remarks

3D effects require four colors, highlight, face, shadow, and dark shadow.   This property controls the dark shadow color.

Data Type

Color

ThreedFaceColor Property

See Also

Description

Returns/sets the listbox's 3D face color.

Syntax

object.**ThreedFaceColor** [= *color*]

The syntax of the **ThreedFaceColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression that specifies the color to use for the face of the item.

Remarks

3D effects require four colors, highlight, face, shadow, and dark shadow.   This property controls the face color.   In the standard windows 3D color-scheme, this is grey --- the same color used on the face of the standard command button.

Data Type

Color

ThreedHiliteColor Property

See Also

Description

Returns/sets the listbox's 3D highlight color.

Syntax

object.**ThreedHiliteColor** [= *color*]

The syntax of the **ThreedHiliteColor** property has these parts:

Part	**Description**
object	A List/X+ control.
color	A color expression that specifies the color to use for the lightest portion of the shadow.

Remarks

3D effects require four colors, highlight, face, shadow, and dark shadow.   This property controls the lightest color.   In the standard windows 3D color-scheme, this is white.

Data Type

Color

ThreedShadowColor Property

See Also

Description

Returns/sets the listbox's 3D shadow color.

Syntax

object.**ThreedShadowColor** [= *color*]

The syntax of the **ThreedShadowColor** property has these parts:

Part	Description
object	A List/X+ control.
color	A color expression that specifies the color to use for the light portion of the shadow.

Remarks

3D effects require four colors, highlight, face, shadow, and dark shadow.   This property controls the shadow color.   Normally there are two lighter colors and one darker one, but you can reverse the settings or try other combinations.

Data Type

Color

TopIndex Property

Description

Returns/sets the index number of the topmost item in the list.

Syntax

object.**TopIndex** [= *long*]

The syntax of the **TopIndex** property has these parts:

Part	**Description**
object	A List/X+ control.
long	A long integer specifying an item to display at the top of the list.

Remarks

TopIndex is used to return/set the index of the topmost item displayed in the listbox.   TopIndex must be between zero and ListCount - 1.

Data Type

Long

Column Object UserResizeEnabled Property

See Also

Description

The user can resize a column when this property is set to True.

Syntax

object.**UserResizeEnabled** [= *boolean*]

The syntax of the **UserResizeEnabled** property has these parts:

Part	Description
object	Required. A Column object.
boolean	A boolean expression that determines if the user may resize this column.

Remarks

When a column's UserResizeEnabled property is set to True, the user may resize the column by dragging the separator between the column and the column to its right.   The MinWidth and MaxWidth properties can be used to restrict the range of sizes a user may choose.

Data Type

Boolean

Version Property

Description

Returns the version of the control.

Syntax

object.**Version**

The syntax of the **Version** property has these parts:

Part	**Description**
object	A List/X+ control.

Remarks

This property holds the current version of the control. It is read-only and available at both design-time and run-time.

Data Type

String

VertDivForeColor Property

See Also

Description

Returns/sets the vertical divider foreground color.

Syntax

object.**VertDivForeColor** [= *color*]

The syntax of the **VertDivForeColor** property has these parts:

Part	Description
object	A List/X+ control.
color	A color expression that specifies the color of the vertical dividers.

Remarks

This color is used to draw the vertical dividers, if any.

Data Type

Color

VertDividers Property

Description

Controls the display of vertical lines between columns.

Syntax

object.**VertDividers** [= *effects*]

The syntax of the **VertDividers** property has these parts:

Part	Description
object	A List/X+ control.
effects	An integer that specifies the style of the vertical dividers.

Remarks

This property supports the following values:

Constant	Value	Description
DividerEffectNone	0	None
DividerEffectSingle	1	Single line
DividerEffectDash	2	Dash
DividerEffectDot	3	Dot
DividerEffectDashDot	4	Dash-Dot
DividerEffectDashDotDot	5	Dash-Dot-Dot

Column Object Visible Property

See Also

Description

Column visible when set to True.

Syntax

object.**Visible** [= *boolean*]

The syntax of the **Visible** property has these parts:

Part	Description
object	Required. A Column object.
boolean	A boolean expression that determines if the column is visible.

Remarks

Enables/disables column visibility.   Note that if a column's width is zero, the column is not displayed, regardless of the setting of the Visible property.

Data Type

Boolean

Column Object Width Property

See Also

Description

Returns/sets column width.

Syntax

object.**Width** [= *long*]

The syntax of the **Width** property has these parts:

Part	**Description**
object	Required. A Column object.
long	A long integer that specifies the column width.

Remarks

If a column's width is zero, the column is not displayed.

Data Type

Integer

Getting Custom Controls Written

If you or your organization would like to have custom controls written, you can contact us at the following:

Mabry Software, Inc.
503 316th Street Northwest
Stanwood, WA   98292

Phone: 360-629-9278
Fax: 360-629-9278

Internet: mabry@mabry.com

You can also contact Zane Thomas.   He can be reached at:

Zane Thomas
Post Office Box 121
Indianola, WA   98342

Internet: zane@mabry.com

Licensing Information

Legalese Version

Mabry Software grants a license to use the enclosed software to the original purchaser. Copies may be made for back-up purposes only. Copies made for any other purpose are expressly prohibited, and adherence to this requirement is the sole responsibility of the purchaser.

Customer written executable applications containing embedded Mabry products may be freely distributed, without royalty payments to Mabry Software, provided that such distributed Mabry product is bound into these applications in such a way so as to prohibit separate use in design mode, and that such Mabry product is distributed only in conjunction with the customers own software product. The Mabry Software product may not be distributed by itself in any form.

Neither source code for Mabry Software products nor modified source code for Mabry Software products may be distributed under any circumstances, nor may you distribute .OBJ, .LIB, etc. files that contain our routines. This control may be used as a constituent control only if the compound control thus created is distributed with and as an integral part of an application. Permission to use this control as a constituent control does not grant a right to distribute the license (LIC) file or any other file other than the control executable itself. This license may be transferred to a third party only if all existing copies of the software and its documentation are also transferred.

This product is licensed for use by only one developer at a time. Mabry Software expressly prohibits installing this product on more than one computer if there is any chance that both copies will be used simultaneously. This restriction also extends to installation on a network server, if more than one workstation will be accessing the product. All developers working on a project which includes a Mabry Software product, even though not working directly with the Mabry product, are required to purchase a license for that Mabry product.

This software is provided as is. Mabry Software makes no warranty, expressed or implied, with regard to the software. All implied warranties, including the warranties of merchantability and fitness for a particular use, are hereby excluded.

MABRY SOFTWARE'S LIABILITY IS LIMITED TO THE PURCHASE PRICE. Under no circumstances shall Mabry Software or the authors of this product be liable for any incidental or consequential damages, nor for any damages in excess of the original purchase price.

To be eligible for free technical support by telephone, the Internet, CompuServe, etc. and to ensure that you are notified of any future updates, please complete the enclosed registration card and return it to Mabry Software.

English Version

We require that you purchase one copy of a control per developer on a project. If this is met, you may distribute the control with your application royalty free. You may never distribute the LIC file. You may not change the product in any way that removes or changes the requirement of a license file.

We encourage the use of our controls as constituent controls when the compound controls you create are an integral part of your application. But we don't allow distribution of our controls as constituents of other controls when the compound control is not part of an application. The reason we need to have this restriction is that without it someone might decide to use our control as a constituent, add some trivial (or even non-trivial) enhancements and then sell the compound control. Obviously there would be little difference between that and just plain reselling our control.

If you have purchased the source code, you may not re-distribute the source code either (nor may you copy it into your own project). Mabry Software retains the copyright to the source code.

Your license is transferable. The original purchaser of the product must make the transfer request. Contact us for further information.

The sample versions of our products are intended for evaluation purposes only. You may not use the sample version to develop completed applications.