VSVBX 3.0
The VideoSoft Custom Control Library

Table of Contents

Introduction

Welcome to VSVBX 3.0, the VideoSoft Custom Control Library. If you are upgrading from a previous version, make sure to look in the "New Features in VSVBX 3.0" section.

The VideoSoft Controls were designed to save developers from writing tedious, repetitive, error-prone code. The controls are innovative and efficient. They are distributed as a single VBX to make installation easier.

Our distribution policy is as innovative as the controls themselves. We think buying software before trying it out is like buying a car without test-driving it. So we want a lot of people to get copies of VSVBX and use it on a trial basis. Executables created with unlicensed copies will display a VideoSoft banner before running, to remind people to get a license. We think most people who try VSVBX will like it, use it, and will eventually buy a license.

We hope you'll like VSVBX. If you have suggestions and ideas for new features or new controls, call us or write.

VideoSoft
2625 Alcatraz Avenue, Suite 271
Berkeley, CA 94705
(510) 547-7295

Control Summary

Icon	Name	Object	Description
	Elastic	VSElastic	Smart containers that resize themselves and their child controls, automatically create labels and 3-D frames for its child controls, and can also be used as progress indicators and labels.
	IndexTab	VSIndexTab	Allow you to group controls by subject, using a familiar notebook metaphor.
	Awk	VSAwk	Parsing engine named and patterned after the popular Unix utility.

Licensing

If you haven't yet purchased your VSVBX license, but are ready to do so, fill out the registration form on the last page of this document and send it in. Or, if you can't wait, call us at (510) 5477295 with your credit card number ready.

When you purchase your VSVBX license, you will get the following:

1. The latest release of the software.
2. Printed documentation.
3. Notification of new releases and special upgrade offers.
4. Sample projects that show how to use the controls.
5. Product support through CompuServe, mail, or phone.
6. The VSVBX license files.
7. The satisfaction of helping us develop high-quality software for you.

Product Support

Product support for VSVBX is available to licensed users through the following channels:

CompuServe: CIS 71552,3052

Mail: VideoSoft
2625 Alcatraz Avenue, Suite 271
Berkeley, California 94705

Phone: (510) 547-7295

New Features in VSVBX 3.0

This section summarizes the new features in VSVBX 3.0. For details on each new feature, check the main body of the documentation. Most of these enhancements were added in response to user requests, and we are grateful for their input.

VSVBX 3.0 is a vast improvement over previous versions. It is easier to use, faster, and offers incredible new functionality. It also loads and saves forms much faster than previous versions.

VSVBX 3.0 supports context-sensitive help. Press F1 from Visual Basic while one of the VideoSoft controls is active, and you'll get a summary screen with all properties, methods, and events for the control. (The help file must reside in the same diretory as the VBX file.)

We have also improved the documentation to make it more accurate, complete, and accessible to new users. Check out the "Getting Started" chapters on the IndexTab and Elastic.

Elastic 3.0

The new Elastic features faster and more accurate child positioning, better handling of graphical child controls, and is also easier to use than before. You no longer have to tell Elastics to resize themselves when the form is resized. The new Elastics are smarter and resize themselves automatically.

New Properties

AccessKey lets you use the Elastic as a label, with access key (&-key) functionality.

BevelChildren lets you determine which controls within the Elastic receive a border.

MaxChildSize and **MinChildSize** extend your control over automatic child resizing.

WordWrap gives you more control over the way the Elastic's Caption is drawn.

TagWidth makes it easier to design data-entry forms. It allows you to use the Tag property of the Elastic's children as labels. By doing this, you can eliminate a number of Label controls and make sure that each label always stays next to its control when you move the control about the form.

Hwnd and **HelpContextID** are standard properties that were not available in previous versions.

New Settings

The **AutoSizeChildren** property has three new settings: "5 Elastics Horizontally",
"6 Elastics Vertically", and "7 Proportional."

With the *Proportional* setting, all controls within an Elastic are resized proportionally when the screen is resized. This dramatically simplifies the design of some forms and makes the Elastic much easier to use. It also allows you to use fewer Elastics in your forms.

IndexTab 3.0

The new IndexTab features multiple access keys (&-keys) in the caption and new tab styles. Redrawing is faster and flicker-free, tab scroling is smoother, and the new tabs also look better.

Using the AutoSwitch property is easier now. You no longer have to worry about keeping track of the Z-Order of the controls: the new IndexTab does that for you automatically, based on the position of the controls.

New properties

ShowFocusRect lets you control whether the IndexTab should draw a focus rectangle around the current tab when it has the focus.

TabCaption is a property array that allows you to easily change the caption of individual tabs. You can also use it to add new tabs or to remove existing ones.

TabsPerPage gives you extra control over the way tabs are displayed. You can use it to make all the tabs the same size and to make them spread over the available space.

BackSheets controls how the IndexTab draws the edges of hidden sheets.

TabOutlineColor gives you extra control over the way tabs are displayed.

Hwnd and **HelpContextID** are standard properties that were not available in previous versions.

New Settings

The **Style** property has two new settings: "Chamfered" and "Chamfered 3 D."

The **Position** property has a new setting: "Right face in", which draws tabs along the right edge of the control with text facing into the IndexTab, rather than out.

Awk 3.0

The new Awk has a larger buffer for parsing really long strings (up to 64k versus 256 bytes in previous versions), the ability to read binary files, and improvements to the way the FS property works.

New Properties

FileType determines whether files are to be opened in text or binary mode.

MatchQuotes makes it easier to parse quote and comma delimited records in files created by dBase and spreadsheets.

PosAt returns the starting position of a field.   It was designed to be used in conjunction with the **FieldAt** property.

New Settings

The FS property now behaves more consistently with the original Awk FS built-in variable.   See the description of the FS property for details.

The Elastic Control

Description The VideoSoft Elastic control is a versatile smart container. It can save you hundreds of lines of VB code by allowing you to:

1. Automatically resize the Elastic to the left, right, top, bottom, or to fill its container, be it a form or another control.

2. Automatically resize the Elastic's child controls, evenly or unevenly, vertically, horizontally, or proportionally.

3. Label child controls based on their Tag property, instead of using several Label controls.

4. Allow the user to resize controls inside the Elastic at run time, using the mouse (splitter bars).

5. Create multi-line labels, 3-D gauges, or both at the same time in a single Elastic.

6. Give plain controls a 3-D look.

File Name VSVBX.VBX

Object Type VSElastic

Remarks The maximum number of children an Elastic can resize is 30. If you need more, use nested Elastics. Or you may be able to cut down the number of children you need by using the **TagWidth** property.

If you set an Elastic's **BorderWidth** and **ChildSpacing** properties to 0 and fill it with other controls, it may become impossible to select it with the mouse. You can still select it either by cycling through the controls with the tab key or by choosing it from the drop-down list box on top of the Properties Window.

Changing certain properties in the Text and List controls forces them to be destroyed and recreated. If they are inside an Elastic with the **AutoSizeChildren** property set to uneven spacing, other controls will expand to fill the Elastic when the Text or List controls are destroyed; when they are recreated, there will be no room for them, so their size will be reset to zero.

Elastic Summary

Default Property: Caption

Properties

(About)	*AccessKey [3]	*Align
*AutoSizeChildren	BackColor	*BevelChildren [3]
*BevelInner	*BevelInnerWidth	*BevelOuter
*BevelOuterWidth	*BorderWidth	Caption
*CaptionPos	*ChildSpacing	DragIcon
DragMode	Enabled	*FloodColor
*FloodDirection	*FloodPercent	FontBold
FontItalic	FontName	FontSize
FontStrikethru	FontUnderline	ForeColor
Height	HelpContextID [3]	hWnd [3]
Index	Left	*MaxChildSize [3]
*MinChildSize [3]	MousePointer	Name
Parent	*Splitter	TabIndex
TabStop	Tag	*TagWidth [3]
Top	Visible	Width
*WordWrap [3]		

Events

Click	DblClick	DragDrop
DragOver	GotFocus	KeyDown
KeyPress	KeyUp	LostFocus
MouseDown	MouseMove	MouseUp
*RealignFrame	*ResizeChildren	

Methods

Move	Refresh	SetFocus
ZOrder		

AccessKey Property [3.0]

Description Determines whether the Elastic should show and process access keys. Access keys are created by preceding characters in the **Caption** with an ampersand (&).

Usage [*form.*]*VSElastic*.**AccessKey** [= {**True**|**False**}]

Remarks If this property is set to True, the user can use ALT-key combinations to move the focus in a form. If it is set to False, ampersands are displayed as regular text in the Caption and ALT-key combinations are ignored.

This property affects the **Caption** and also the tag labels it the **TagWidth** property is set to a non-zero value.

Default Value False

Data Type Boolean

Align Property

Description Sets or returns the automatic alignment setting for the Elastic.

Usage [*form.*]*VSElastic*.**Align** [= *setting%*]

Remarks Valid options for this property are:

0 - No automatic alignment
1 - Top
2 - Bottom
3 - Left
4 - Right
5 - Fill Container

The Elastic's **Align** property is similar to the standard Align property, except for two things:

The Elastic's Align has three additional settings: Left, Right, and Fill Container.

The Elastic's Align works even when the Elastic is inside another control. The standard VB Align property only works when the control is placed on the form.

Default Value 0 - No automatic alignment

Data Type Integer

AutoSizeChildren Property

Description Determines how the Elastic resizes the controls inside it.

Usage [*form.*]*VSElastic*.**AutoSizeChildren** [= *setting%*]

Remarks This property is useful at design time, to create forms that look professional, and at run time, to make all controls inside the Elastic resize themselves when the form is resized.

Valid options for this property are:

0 - No automatic child sizing
1 - Even Horizontally
2 - Uneven Horizontally
3 - Even Vertically
4 - Uneven Vertically
5 - Elastics Horizontally [3.0]
6 - Elastics Vertically [3.0]
7 - Proportional

The *Even Horizontally and Even Vertically* options make all controls in the Elastic have the same width or height, respectively, and spread them evenly across the Elastic.

The *Uneven Horizontally* and *Uneven Vertically* options stretch only the top control inside the Elastic (highest Z-Order) and spread all child controls across the Elastic.

The *Elastics Horizontally* and *Elastics Vertically* options stretch only other Elastics inside the Elastic and spread all child controls. They can be used to center controls and to form clusters of controls inside an Elastic.

The *Proportional* setting keeps the relative size and position of all child controls constant when the Elastic is resized. While this setting is active, all controls in the Elastic become locked; you can't move them because the Elastic will put them back where they were. To change the layout of the form, you first have to set the AutoSizeChildren property to *None.*

The maximum number of children an Elastic can resize is 30. If you need more, use nested Elastics.

Default Value 0 - None

Data Type Integer

BevelInner, BevelOuter Property

Description	Sets or returns the 3-D appearance of the Elastic and of the controls inside it.
Usage	[*form.*]*VSElastic*.**BevelInner** [= *setting%*] [*form.*]*VSElastic*.**BevelOuter** [= *setting%*]
Remarks	Valid options for this property are:

0 - None
1 - Raised
2 - Raised outlined
3 - Inset
4 - Inset outlined

Default Value	BevelOuter:	2 - Raised Outlined
	BevelInner:	3 - Inset
Data Type	Integer	

BevelInnerWidth, BevelOuterWidth Property

Description	Sets or returns the thickness of the bevels defined by the **BevelInner** and **BevelOuter** properties. Units are pixels.
Usage	[*form.*]*VSElastic*.**BevelInnerWidth** [= *value%*] [*form.*]*VSElastic*.**BevelOuterWidth** [= *value%*]
Remarks	The thickness of the bevels is limited by the Elastic's **BorderWidth.**
Default Value	BevelOuterWidth: 2 BevelInnerWidth: 1
Data Type	Integer

BevelChildren Property [3.0]

Description	Determines whether the Elastic should draw 3-D frames around all its child controls or only around specified types of controls.
Usage	[*form.*]*VSElastic*.**BevelChildren** [= *setting%*]
Remarks	Valid setttings for this property are:

0 - All
1 - No Graphical
2 - No Elastics
3 - No Graphical or Elastics

This property adds control over the look of forms that contain labels and edit controls. By setting **BevelChildren** to a value other than zero, you can make the labels look flat and the edit controls look raised or inset.

Default Value	0 - All
Data Type	Integer

BorderWidth Property

Description	Sets or returns the width of the border around the child controls and the Elastic.
Usage	[*form.*]*VSElastic*.**BorderWidth** [= *setting%*]
Remarks	The border width is measured in pixels.
Default Value	6
Data Type	Integer

CaptionPos Property

Description	Sets or returns the position of the Elastic's caption. Also sets the position of the tag labels, if they are enabled (see the **TagWidth** property).
Usage	[*form.*]*VSElastic*.**CaptionPos** [= *setting%*]
Remarks	Valid options for this property are:

0 - Left Top
1 - Left Center
2 - Left Bottom
3 - Center Top
4 - Center Center
5 - Center Bottom
6 - Right Top
7 - Right Center
8 - Right Bottom

Default Value	1 - Left Center
Data Type	Integer

ChildSpacing Property

Description Sets or returns the spacing between child controls inside the Elastic.

Usage [*form.*]*VSElastic*.**ChildSpacing** [= *setting%*]

Remarks The child spacing is measured in pixels.

This property is only relevant when the **AutoSizeChildren** property is set to a non-zero value.

Default Value 6

Data Type Integer

FloodColor Property

Description	Sets or returns the color used to fill the Elastic when it is used as a progress indicator.
Usage	[*form.*]*VSElastic*.**FloodColor** [= *color&*]
Remarks	This property is only relevant when the **FloodDirection** property is set to a non-zero value.
Default Value	Red
Data Type	Long

FloodDirection Property

Description	Sets or returns the direction used to fill the Elastic when it is used as a progress indicator.
Usage	[*form.*]*VSElastic*.**FloodDirection** [= *setting%*]
Remarks	Valid settings for this property are:

0 - None
1 - Fill Right
2 - Fill Left
3 - Fill Up
4 - Fill Down

This property is only relevant when the **FloodDirection** property is set to a non-zero value.

Default Value	0 - None
Data Type	Integer

FloodPercent Property

Description	Sets or returns the percentage of the Elastic that should be flooded when it is used as a progress indicator.
Usage	[*form.*]*VSElastic*.**FloodPercent** [= *setting%*]
Remarks	Setting this property to a value smaller than 0 or greater than 100 will not cause any errors.

This property is only relevant when the **FloodDirection** property is set to a non-zero value.

Default Value	Zero
Data Type	Integer

MaxChildSize Property [3.0]

Description Sets or returns the maximum size for the Elastic's child controls. The size is measured in Twips and may correspond to the width or height of the child control, depending on the setting of the **AutoSizeChildren** property.

Usage [*form.*]*VSElastic*.**MaxChildSize** [= *value&*]

Remarks The maximum size limit only applies to the child controls being resized by the Elastic. If **AutoSizeChildren** is set to uneven Spacing, for example, only the top child is affected.

This property is useful to prevent controls inside an Elastic from spreading too much. Command buttons, for example, are harder to use if they are too far apart from each other.

Setting this property to 0 disables it.

Default Value Zero

Data Type Long

MinChildSize Property [3.0]

Description Sets or returns the minimum size for the Elastic's child controls. The size is measured in Twips and may correspond to the width or height of the child control, depending on the setting of the **AutoSizeChildren** property.

Usage [*form.*]*VSElastic*.**MinChildSize** [= *value&*]

Remarks The minimum size limit only applies to the child controls being resized by the Elastic. If **AutoSizeChildren** is set to uneven Spacing, for example, only the top child is affected.

This property is useful to prevent controls inside an Elastic from getting too narrow. It may be better, for example, to show only a few command buttons with readable captions than to show all of them truncating the captions.

This property can be used in conjunction with the **Splitter** property to prevent users from making a child control too small.

Setting this property to 0 disables it.

Default Value Zero

Data Type Long

RealignFrame Event

Description	Fired after the Elastic is resized.
Syntax	**Sub** *CtlName*_**RealignFrame ()**

ResizeChildren Event

Description	Fired after the Elastic's children are resized.
Syntax	**Sub** *CtlName*_**ResizeChildren ()**

Splitter Property

Description Determines whether the user is allowed to resize Elastic child windows by dragging the bar between two child windows.

Usage [*form.*]*VSElastic*.**Splitter** [= {**True**|**False**}]

Remarks If the **Splitter** property is set to **True**, the Elastic will monitor the mouse at run time. Whenever the mouse passes over the space between controls inside the Elastic, the pointer changes to a resize icon and the user is allowed to drag the border between the adjacent controls to a new location.

To use the **Splitter** property, the **AutoChildSpacing** property must be set to *Uneven Vertical* or *Uneven Horizontal*, and the **TafWidth** property must be set to zero.

Default Value False

Data Type Boolean

TagWidth Property [3.0]

Description This property allows you to use the Tag property of the Elastic's children as labels, instead of using label controls. This makes forms easier to create and can also saves lots of labels.

Usage [*form.*]*VSElastic*.**TagWidth** [= *setting&*]

Remarks Tag labels are always placed to the left of the control being labeled, and their height is the same as that of the control being labeled. The width of the labels is controlled by the **TagWidth** property, as explained below.

Positive **TagWidth** values are interpreted as the width of the tag labels, measured in twips. For example, if you set the **TagWidth** to 1000, the tag labels will always be 1000 twips wide.

Negative **TagWidth** values are interpreted as a percentage of the width of the control being labeled. For example, if you set **TagWidth** to 50, the tag labels will always be half as wide as the controls being labeled.

Setting **TagWidth** to zero disables the tag labels.

Label justification is controlled by the **CaptionPos** property. The appearance and behavior of the tag labels is also affected by the **AccessKey, WordWrap,** and **Font...** properties.

Default Value Zero

Data Type Long

WordWrap Property [3.0]

Description This property determines whether the caption text should be automatically broken between words if a word would extend past the edge of the control. Return characters will also break the caption.

Usage [*form.*]*VSElastic*.**WordWrap** [= {**True**|**False**}]

Remarks If an Elastic has room for only one line of text and a caption with several words, word wrapping may hide parts of the caption. In such cases, setting this property to False may improve readability.

This property also affect the tag labels, if they are enabled (see the **TagWidth** property).

Default Value True

Data Type Boolean

The IndexTab Control

Description The VideoSoft IndexTab control allows you to group controls by subject, using a familiar notebook metaphor.

File Name VSVBX.VBX

Object Type VSIndexTab

Remarks To use the IndexTab control, follow these steps:

1. Draw the IndexTab.
2. Set the **Caption** property to create as many tabs as you need.
3. Create one Picture Box or Elastic per tab, inside the IndexTab control.
4. Create the controls inside each container.
5. Set other IndexTab properties as desired.

Some applications need to change the text in the tabs at run time. You can do this easily using the **TabCaption** property. For example, the following code changes the caption of the first tab in a IndexTab control:

```
VSIndexTab.TabCaption(0) = "new caption"
```

IndexTab Summary

Default Property: **CurrTab**

Properties:

(About)	*AutoScroll	*AutoSwitch
BackColor	*BackSheets [3]	*BackTabColor
BorderStyle	*Caption	*ClientLeft
*ClientHeight	*ClientTop	*ClientWidth
*CurrTab	DragIcon	DragMode
Enabled	*FirstTab	FontBold
FontItalic	FontName	FontStrikethru
FontUnderline	ForeColor	*FrontTabColor
Height	HelpContextID [3]	Hwnd [3]
Index	Left	MousePointer
Name	*NumTabs	*Position
*ShowFocusRect [3]	*Style	*TabCaption [3]
TabIndex	*TabOutlineColor [3]	*TabsPerPage [3]
TabStop	Tag	Top
Visible	Width	

Events

*Click	DblClick	DragDrop
DragOver	GotFocus	KeyDown
KeyPress	KeyUp	LostFocus
MouseDown	MouseMove	MouseUp
*Scroll		

Methods

Move	Refresh	SetFocus
ZOrder		

AutoScroll Property

Description Determines whether the control should automatically scroll the list of tabs so that the current one is always visible.

Usage [*form.*]*VSIndexTab*.**AutoScroll** [= {**True**|**False**}]

Remarks If this property is set to True, the user can scroll the Tabs using the arrow keys or clicking on a partially hidden tab.

You can find out when scrolling happened by checking the **Scroll** event.

Default Value True

Data Type Boolean

AutoSwitch Property

Description	Determines whether the control should automatically resize, show, and hide child windows when the current tab changes.
Usage	[*form.*]*VSIndexTab*.**AutoSwitch** [= {**True**|**False**}]
Remarks	To use the **AutoSwitch** feature, you need to design your IndexTab so that it has one child control per tab, including comment tabs.

When a tab is clicked, the IndexTab chooses which child control to show based on the children's Z-Order. Prior to Version 3.0, you had to sort the children Z-Order manually. In Version 3.0, the IndexTab automatically sorts the children based on their position. The leftmost child control is always sent to the back, and corresponds to the first tab; the rightmost child control is always brought to the front, and corresponds to the last tab.

This property only acts at run time.

Default Value	True
Data Type	Boolean

BackSheets Property [3.0]

Description	Defines the appearance of the back sheets behind the current tab.
Usage	[*form.*]*VSIndexTab*.**BackSheets** [= *setting%*]
Remarks	Valid options for this property are:

0 - None
1 - Shadow
2 - Sheets

Default Value	2 - Sheets
Data Type	Integer

BackTabColor Property

Description	Defines the color of all non-selected tabs.
Usage	[*form.*]*VSIndexTab*.**BackTabColor** [= *color&*]
Remarks	Use this property in conjunction with **BackColor** and **FrontTabColor** to define the appearance of the control.
Default Value	Light gray
Data Type	Long

Caption Property

Description	Determines the number of tabs and the text they contain.
Usage	[*form.*]*VSIndexTab*.**Caption** [= *string$*]
Remarks	Use the pipe character ("|") to separate options. Use a dash ("-") followed by spaces to create an invisible tab between regular tabs. Each tab may have an access key (&-key) to allow for fast tab switching.
Data Type	String

Click Event

Description Fired when the current tab changes either through code, mouse, or keyboard action.

Syntax **Sub** *CtlName*_**Click ()**

ClientLeft, ClientTop, ClientWidth, ClientHeight Properties

Description Return the internal dimensions of the IndexTab excluding the rectangle used to show the tabs themselves. Units are Twips.

Usage [*form.*]*VSIndexTab*.**ClientWidth**

Remarks These properties are useful if you want to size frames or picture controls so that they occupy the whole client area of the control.

You only need to use these properties if the **AutoSwitch** property is set to False.

These properties are read-only.

Data Type Single

CurrTab Property

Description Sets or returns the number of the currently selected tab.

Usage [*form.*]*VSIndexTab*.**CurrTab** [= *setting%*]

Remarks The first tab is numbered zero. Invisible tabs, used to separate regular tabs, are also counted.

Setting this property to a negative value or to a value greater than the number of tabs causes all tabs to be de-selected. No error occurs.

Changes to this property fire the **Click** event.

Default Value Zero

Data Type Integer

FirstTab Property

Description Sets or returns the number of the first tab to be displayed.

Usage [*form.*]*VSIndexTab*.**FirstTab** [= *setting%*]

Remarks The first tab is numbered zero. Invisible tabs, used to separate regular tabs, are also counted.

This property is useful if you want to implement scrolling in addition to that provided by the **AutoScroll** property.

Changes to this property fire the **Scroll** event.

Default Value Zero

Data Type Integer

FrontTabColor Property

Description	Defines the color of current tab and front page of the control.
Usage	[*form.*]*VSIndexTab*.**FrontTabColor** [= *color&*]
Remarks	Use this property in conjunction with **BackColor** and **BackTabColor** to define the appearance of the control.
Default Value	White
Data Type	Long

NumTabs Property

Description	Returns the number tabs currently defined.
Usage	[*form.*]*VSIndexTab*.**NumTabs**
Remarks	This property is read-only. IndexTabs automatically setst it whenever you change the **Caption** or **TabCaption** properties.
Data Type	Integer

Position Property

Description	Sets or returns the position of the tabs within the control.
Usage	[*form.*]*VSIndexTab*.**Position** [= *setting%*]
Remarks	Valid options for this property are:

0 - Top
1 - Bottom
2 - Left
3 - Right
4 - Right, face in [3].

The difference between settings 3 and 4 is in the text orientation. With setting 3, the text is written from the bottom up; with setting 4, the text is written from the top down (facing the interior of the IndexTab).

Windows only supports vertical text with some fonts. If you set the **Position** property to 2 or 3, the IndexTab will automatically convert the font into the nearest TrueType equivalent.

Text written in the vertical direction does not display acelerator keys correctly. This is a Windows limitation.

Default Value	1 - Bottom
Data Type	Integer

Scroll Event

Description Fired when the **FirstTab** property changes either through code, mouse, or keyboard action.

Syntax **Sub** *CtlName*_**Scroll ()**

ShowFocusRect Property

Description	Determines whether the control should show a focus rectangle around the current tab when it has the focus.
Usage	[*form.*]*VSIndexTab*.**ShowFocusRect** [= {**True**|**False**}]
Default Value	True
Data Type	Boolean

Style Property

Description	Sets or returns the style used to draw the tabs.
Usage	[*form.*]*VSIndexTab*.**Style** [= *setting%*]
Remarks	Valid options for this property are:

0 - Slanted lines;
1 - Slanted lines plus 3D effect;
2 - Rounded corners
3 - Rounded corners plus 3D effect
4 - Chamfered corners
5 - Chamfered corners plus 3D effect

Default Value	0 - Slanted Lines
Data Type	Integer

TabCaption Property [3.0]

Description Sets or retrieves the caption for an individual tab.

Usage [*form.*]*VSIndexTab*.**TabCaption(***index%***)** [= *string$*]

Remarks Use this property to dynamically change tabs in the IndexTab.

You can create new tabs by assigning a string with embedded pipe characters to a TabCaption. You can delete tabs by assigning them an empty string.

Trying to access a tab smaller than 0 or greater than **NumTabs**-1 will generate an invalid index error.

This property only acts at run time.

Example

```
VSIndexTab.Caption = "First|Second|Third"
debug.print VSIndexTab.TabCaption(0)
 First

VSIndexTab.TabCaption(0) = "First|New Second"
debug.print VSIndexTab.Caption
 First|New Second|Second|Third

VSIndexTab.TabCaption(0) = ""
debug.print VSIndexTab.Caption
 New Second|Second|Third
```

Data Type String array

TabOutlineColor Property [3.0]

Description	Defines the color to be used for the outline of the tabs.
Usage	[*form.*]*VSIndexTab*.**TabOutlineColor** [= *color&*]
Remarks	This property is mainly useful when one of the 3-D styles is selected. White outlines enhance the 3-D effect, while dark gray or black outlines give the control a more subtle look.
Default Value	Black
Data Type	Long

TabsPerPage Property [3.0]

Description	Determines the number of tabs that should be displayed in the control.
Usage	[*form.*]*VSIndexTab*.**TabsPerPage** [= *n%*]
Remarks	If this property is set to 0, the size of each tab is determined by the length of its caption. If this property is greater than zero, all tabs are drawn with the same size, so that the total number of tabs visible at any time is equal to the value of the property.
Default Value	Zero
Data Type	Integer

The Awk Control

Description The VideoSoft Awk control allows you to quickly scan and parse text files. It is the ideal tool for doing simple data manipulation -- changing its format, checking its validity, retrieving items, generating reports, and the like -- without having to write a lot of code to scan and parse files.

File Name VSVBX.VBX

Object Type VSAwk

Remarks The VideoSoft Awk control is similar to the original AWK language in concept and purpose, but it is substantially different in syntax, since Awk is embedded in Visual Basic. The most apparent omission is a pattern matching mechanism, which is provided by VB's **Like** operator.

Although Awk was designed primarily for scanning and parsing files, you can use it to parse anything you want. For example, the following routine parses a file specification into drive, path, and file name:

```
Sub ParseFileSpec (FileSpec as String)

  ' set line and field separator properties
  VSAwk = FileSpec
  VSAwk.FS = "\"

  ' get the file name and clear its field
  Debug.Print "file "; VSAwk.F(VSAwk.NF)
  VSAwk.F(VSAwk.NF) = ""

  ' get the drive and creal its field
  Debug.Print "drive "; VSAwk.F(1)
  VSAwk.F(1) = ""

  ' whatever is left is the dir
  Debug.Print "dir "; VSAwk
End Sub
```

Awk can read lines up to 64k characters long while scanning files. If a line exceeds this limit, Awk truncates the line and generates an **Error** event.

Awk is different from most other custom controls in that it does not generate run time errors. Instead, it fires the **Error** event and sets its own internal **Error** property. Therefore, Awk ignores errors by default. You can trap errors either by trapping the **Error** event or by testing the **Err** property immediately after using any other Awk property.

Awk Summary

Default Property: **L**

Properties

(About)	*Action	*Case
*CurrPos	*Error	*F
*FieldAt	*FileName	*FileType
*FS	Index	*L
*MatchQuotes [3]	Name	*NF
*PercentDone	*PosAt [3]	*RN
Tag		

Events

*Begin	*End	*Error
*Scan		

Action Property

Description Specifies an action to be performed by the Awk control.

Usage [*form.*]*VSAwk*.**Action** [= *action%*]

Remarks Valid actions are:

0 - Scan File
1 - Next Line
2 - Close File

The *Scan File* action causes Awk to take the following actions:

1. Open the file specified by the **FileName** property;
2. Fire the **Begin** event;
3. Read the file one line at a time, parsing each line and firing a **Scan** event for each one;
4. Close the file;
5. Fire the **End** event.

The *Next Line* action causes Awk to read the next line from the current data file. It is often useful in the **VSAwk_Scan** subroutine.

The *Close File* action causes Awk to close the current file and stop scanning it.

Data Type Integer

Begin Event

Description Fired when Awk opens a file for scanning, after **Action** is set to 0 (Scan File).

Syntax **Sub** *CtlName*_**Begin ()**

Case Property

Description	Determines whether Awk should change the case of each line as it is read.
Usage	[*form.*]*VSAwk*.**Case** [= *setting%*]
Remarks	Valid setting for this property are:

0 - No Change
1 - Convert to Upper Case
2 - Convert to Lower Case

Default Value	0 - No Change
Data Type	Integer

CurrPos Property

Description	Sets or returns the position of the current line within the file.
Usage	[*form.*]*VSAwk*.**CurrPos** [= *value&*]
Remarks	This property is useful when you want to save the position of current line so that you can quickly return to it later.
Data Type	Long

End Event

Description Fired when Awk is done scanning a file.

Syntax **Sub** *CtlName*_**End ()**

Error Event

Description	Fired when Awk encounters an error. See the **Error** property for a list of possible errors.
Syntax	**Sub** *CtlName*_**Error ()**

Error Property

Description Returns a code that identifies the last error trapped by Awk.

Usage [*form.*]*VSAwk*.**Error**

Remarks Possible error codes are:

0 - No Error
1 - Can't open file
2 - Line too long
3 - Field too long
4 - Invalid field index set
5 - Invalid field index get

The *Can't open file* error occurs when the file specified by the **FileName** property can't be found.

The *Line too long* error occurs when the file contains lines longer than 64k characters. In this case, Awk truncates the line.

The *Field too long* error occurs when you try to make the current line longer than 64k characters.

The *Invalid field index set* error occurs when you try to set the value of a field that doesn't exist, i.e., smaller than zero or greater than **NF**.

The *Invalid field index get* error occurs when you try to get the value of a field that doesn't exist, i.e., smaller than zero or greater than **NF**.

Data Type Integer

F (Field) Property

Description Sets or returns the value of a field within the current line (record).

Usage *[form.]VSAwk*.**F(***index%***)** [= *value$*]

Remarks Fields are numbered from 1 to NF. As in the Awk language, field 0 corresponds to the whole line (record).

If you try to set or read an invalid field, Awk fires the **Error** event.

If you set a field to a string that contains field separators, the line is automatically re-parsed, and the **NF** property is updated.

Example

```
VSAwk.F(0) = "This is a string"
Debug.Print VSAwk.NF, VSAwk.F(0)
 4       This is a string

Debug.Print VSAwk.F(4)
 string

VSAwk.F(4) = "plain long ASCII string"
Debug.Print VSAwk.NF, VSAwk.F(0)
 7       This is a plain long ASCII string
```

Data Type String array.

FieldAt Property

Description Returns the number of the field at the specified string position.

Usage [*form.*]*VSAwk*.**FieldAt(***position%***)**

Remarks This property was designed to be used with VB's **InStr** function. The first string position is 1.

This property is useful when you want to process a field based on its contents rather than on its position. See also the **PosAt** property, which performs the reverse function.

Example Use the **PosAt** property to find the contents of the field after the string "value".

```
VSAwk = "stuff stuff stuff stuff value data stuff stuff"
debug.print VSAwk.FieldAt(instr(VSAwk, "value"))
 5

debug.print VSAwk.f(5+1)
 data
```

Data Type Integer array.

FileName Property

Description	Sets or returns the name of the file to be scanned by the Awk control.
Usage	[*form.*]*VSAwk*.**FileName** [= *filename$*]
Remarks	If Awk can't find the specified file, it fires the **Error** event.
Data Type	String.

FileType Property

Description Determines whether the file being scanned is a text or binary file. When reading text files, Awk translates carriage return/line feed characters into carriage returns and closes the file when an end-of-file character (ctrl-Z) character is found.

Usage [*form.*]*VSAwk*.**FileType** [= *setting%*]

Remarks Valid settings for this property are:

0 - Text File
1 - Binary File

Data Type 0 - Text File.

FS (Field Separator) Property

Description Sets or returns the string of characters to be used by Awk as field separators.

Usage [*form.*]*VSAwk*.**FS** [= *string$*]

Remarks Each occurrence of any of the characters in the **FS** string marks the beginning of a new field. The only exception is that repeated spaces are equivalent to a single space. See the examples below.

If you change this property, the current string is automatically re-parsed.

Example

```
VSAwk = "This is a string|||with some          pipes in it"
VSAwk.FS = " "
debug.print VSAwk.NF
 8

VSAwk.FS = "|"
debug.print VSAwk.NF
 4

VSAwk.FS = "| "
debug.print VSAwk.NF
 11
```

Default Value " , ->" (space, comma, tab)

Data Type String.

L (Line) Property

Description Sets or returns the value of the current line (record).

Usage [*form.*]*VSAwk*.**L** [= *string$*]

Remarks This is the default property. This property is exactly the same as the **F(0)** property.

Example

```
VSAwk.F(0) = "This is a string"
Debug.Print VSAwk.L
```

This is a string

Data Type String.

MatchQuotes Property [3.0]

Description Determines whether the Awk parser should regard quote-delimited strings as single fields.

Usage [*form.*]*VSAwk*.**MatchQuotes** [= {**True**|**False**}]

Remarks This feature was added to facilitate parsing of quote and comma-delimited files such as those exported by most database and spreadsheet programs.

Double or single quotes are both recognized, but one does not match the other. Unmatched quotes in a line are ignored.

Example

```
VSAwk = "'John Doe', 24, '1244 Main Street', 645-5432"
VSAwk.FS = ", "
VSAwk.MatchQuotes = false
debug.print VSAwk.NF, VSAwk.F(1)
 7    'John

VSAwk.MatchQuotes = true
debug.print VSAwk.NF, VSAwk.F(1)
 4    'John Doe'
```

Default Value False

Data Type Boolean

NF (Number of Fields) Property

Description	Returns the number of fields in the current line (record).
Usage	[*form.*]*VSAwk*.**NF**
Remarks	The number of fields changes whenever the current line (**L** property) or field separators (**FS** property) change.
Data Type	Integer

PercentDone Property

Description	Returns a number between 0 and 100 that indicates how much of the current file has been scanned.
Usage	[*form.*]*VSAwk*.**PercentDone**
Remarks	This property was designed to give you an easy way to report progress while processing long files. It can be used, for example, with the Elastic **FloodPercent** property.
Data Type	Integer

PosAt Property [3.0]

Description Returns the position of the specified field in the current record.

Usage [*form.*]*VSAwk*.**PosAt(***field%***)**

Remarks This property was designed to be used with VB's string functions. The first string position is 1.

This property is useful when you want to process a field based on its contents rather than on its position. See also the **FieldAt** property, which performs the reverse function.

Example Use the **PosAt** property to remove the first four fields from a record.

```
VSAwk = "junk1 junk2 junk3 junk4 data5 data6 data7"
debug.print VSAwk.PosAt(5)
 25

debug.print mid(VSAwk,25)
 data5 data6 data7
```

Data Type Integer array.

RN (Record Number) Property

Description	Returns the number of the current line (record) being scanned by Awk.
Usage	[*form.*]*VSAwk*.**RN**
Data Type	Long

Scan Event

Description Fired for each line in the input file while Awk scans a file. This is where you put the code that does the actual processing.

You may close the current file or get the next input line while in this subroutine by setting the **Action property** to 2 or 3, respectively.

Syntax **Sub** *CtlName*_**Scan ()**

Registration Form

VSVBX: The VideoSoft Custom Control Library

TO:
VideoSoft
2625 Alcatraz Avenue, Suite 271
Berkeley, California 94705

Please register my copy of the VideoSoft Custom Control Library. I am enclosing a check or money order for the amount of:

Single developer	US$45.00
Site License, 10 to 20 developers	US$350.00
Site License, 21 or more developers	US$650.00
Shipping and Handling Domestic	US$6.00
Shipping and Handling International	US$10.00

(CA residents please add 8.5% sales tax).

Name:

Company:

Street:

City, State, ZIP:

Country:

Phone:

Where did you hear about the VideoSoft Custom Controls?