VSVIEW 1.0

VideoSoft Custom Control Library

To learn how to use help, press F1

Introduction
Find out about installation, product support, licensing, registration, and other VideoSoft products.

vsInForm
Customize forms and controls, monitor the clipboard, drag and drop files from the File Manager.

vsPrinter
Print files, text, graphics, and tables with automatic word wrap, headers and footers, multiple columns, and previewing.

vsViewPort
Fit more controls on your windows with virtual scrollable areas.

vsDraw
Create complex images, view them on the screen, copy them to the clipboard, and print them.

Introduction

Welcome to VSVIEW 1.0, a VideoSoft Custom Control Library.

VSVIEW contains four custom controls designed to save you 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 almost as innovative as the controls. We want every Visual Basic programmer to get a copy of VSVIEW and try it for as long as they want. Those who like the product and find it useful (almost everybody, we hope) can buy a license for a reasonable price. The only restriction is that unlicensed copies of VSVIEW display a VideoSoft banner whenever they are loaded, to remind developers to license the product.

We hope you'll like VSVIEW. 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 (phone)
(510) 547-1084 (fax)

Control Summary

Icon	Object	Name	Description
	**vsInForm**	InForm	A control that you can drop into any container to customize its title bar, frame, resizing behavior, and frame buttons. InForm also allows you to monitor the clipboard, drag and drop files from File manager, and more.
	**vsPrinter**	Printer	A much improved printer object with word wrap, headers and footers, multi-column printing, graphics, and multi-page Print Preview capability.
	**vsViewPort**	ViewPort	A control that gives you a scrollable virtual area so you can fit more controls in your windows. Great for implementing Print Preview and programs that look like the Program Manager.
	**vsDraw**	Draw	A versatile drawing control that lets you create complex images, view them on the screen, copy them to the clipboard, or print them. Great for technical drawings, maps, and diagrams.

Installation

To install VSVIEW, just copy the following files to your WINDOWS\SYSTEM directory:

VSVIEW.VBX This file contains the controls. To use VSVIEW from Visual Basic, you must include this file in your project.

VSVIEW.LIC This is the VSVIEW license file. If VSVIEW cannot find this file when it starts running, it displays a VideoSoft banner and waits for the user to click Ok, unless the project was compiled on a machine that had VSVIEW.LIC installed.

VSVIEW.HLP This file contains the VSVIEW on-line help.

If you would rather install VSVIEW in a different directory, thats fine. As long as the help and license files are either in the same directory as the VBX, in the WINDOWS\SYSTEM directory, or in the WINDOWS directory, VSVIEW will find them for you.

Distribution

VSVIEW is royalty-free. You may include copies of the VBX and HLP files with as many copies of as many applications you ship.

You cannot distribute the license file VSVIEW.LIC. And you dont have to: as long as you have the license file installed on your machine, VSVIEW will stamp every application you compile so the banner will not appear when your users run the applications.

If you work with other developers, you may be interested in VideoSofts site licenses. Call us for details.

If you havent yet registered your copy of VSVIEW and would like to do it now, click **HERE** to get a Registration Form.

Product Support

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

CompuServe	CIS 74774,420 or join our forum by typing **GO VIDEOSOFT**
Mail	VideoSoft 2625 Alcatraz Avenue, Suite 271 Berkeley, California 94705
Phone	(510) 547-7295
Fax	(510) 547-1084

Before calling for technical support, please make sure you know what version of VSVIEW you are using. The version number appears in the About box that pops up when you double-click the About property in any of the VSVIEW controls.

Also, please make sure you check the last section of the manual, Hints and Troubleshooting. It contains answers to the most common questions people ask our technical support staff. Maybe you can find your answer there.

InForm Reference

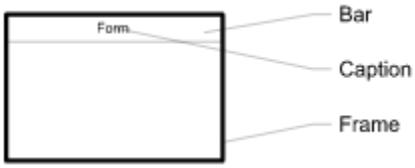
Description InForm allows you to customize the non-client area of its parent form or control. With InForm, you can add a chiseled look to your window borders and caption, add custom buttons, and control window resizing. And because InForm works with any container control, you can add these effects not only to forms, but also to Picture Boxes, Frames, Elastics, IndexTabs, and so on.

InForm also allows you to monitor the clipboard, implement file drag-and-drop, and retrieve information about the environment.

File Name VSVIEW.VBX

Object Type vsInForm

Remarks InForm has many properties that are used to customize its parents non-client area (title bar and window frame.) These properties are divided in three groups: Bar* properties control the title bar, Cap* properties control the caption inside the title bar, and Frame* properties control the window frame. The picture below illustrates what parts of the window these names refer to:



The CustomFrame property determines whether the InForm control should or should not take over managing the non-client area of the InForms parent. If you set this property to True and the parent is a form, the form must not have a menu. Remember also to set the forms ControlBox property to False, or users will be able to use alt-space to show the system menu.

When using InForm, keep in mind that one of Windowss great virtues is that it promotes a standard interface across applications. Generally, it is not a good idea to change this standard. There are many exceptions to this rule, though: Visual Basic itself is a good example.

VBs floating toolbox and palette windows have skinny title bars, to save room. VBs main window can be resized horizontally, but not vertically. With InForm, you can do the same.

InForm Summary

Properties (default: Caption)

(About)
* AcceptFiles
* BarColor
* BarColorInactive
* BarHeight
* BarStyle
* ButtonsLeft
* ButtonsRight
* CapAlign
* CapColor
* CapColorInactive
* CapMultiLine
* CapStyle
Caption
* ClipMon
* CustomFrame
* FileName
FontBold
FontItalic
FontName
FontSize
FontStrike
FontUnder
* FrameButtons
* FrameColor
* FrameColorInactive
* FrameCorners
* FrameSizing
* FrameStyle
* FrameWidth
* FreeGDI
* FreeMemory
* FreeSystem
* FreeUser
Hwnd
Index
Left
* MaxHeight
* MaxWidth
* MinHeight
* MinWidth
Name
* NumFiles
* OnTop
Parent
* PictLeft0
* PictLeft1
* PictLeft2
* PictRight0
* PictRight1
* PictRight2
Tag
Top
* Version

Events

* ClickLButton
* ClickRButton
* DblClickLButton
* DblClickRButton
* DblClickCaption
* DropFile
* Move
* NewClipboardData
* Resize

AcceptFiles Property

Description This property determines whether the user should be able to drop files from the File Manager into the InForms parent window.

Usage [*form.*]*vsInForm*.**AcceptFiles** [= {**True**|**False**}]

Remarks If this property is set to True, the DropFile event is fired when the user drops a file into the InForms parent.

Default Value False

Data Type Boolean

BarColor, BarColorInactive Properties

Description These are the colors used to paint the caption bar of the InForms parent. At run time, InForm chooses whether to use BarColor or BarColorInactive automatically, based on whether the InForms parent is active or not.

Usage [*form.*]*vsInForm*.**BarColor** [= *colorref&*]

Remarks These properties are used only if the CustomFrame property is set to True.

Setting either of these properties to zero causes the system default colors to be used. If you want the caption to be black, set these properties to one (if you simply pick black from the color box, VB will pick color zero and the control will use the default system colors).

Default Value Zero (use system colors)

Data Type Color (Long)

BarHeight Property

Description This property sets or returns the height of the caption bar of the InForms parent, in pixels.

Usage [*form.*]*vsInForm*.**BarHeight** [= *height%*]

Remarks This property is used only if the CustomFrame property is set to True.

Setting this property to zero causes the system default caption height to be used. If you want a window with no caption, set the BarStyle property to None.

Default Value Zero (use system value)

Data Type Integer

BarStyle Property

Description This property determines the appearance of the caption bar of the InForms parent.

Usage [*form.*]*vsInForm*.**BarStyle** [= *setting%*]

Remarks Valid settings for this property are:
0 - None
1 - Classic
2 - NoBorder
3 - Raised
4 - Inset

This property is used only if the CustomFrame property is set to True.

Default Value 1 - Classic

Data Type Integer

ButtonsLeft, ButtonsRight Properties

Description These properties determine how many buttons should be displayed on the left and right-hand side of the InForm parents caption bar. The buttons are similar to the standard Windows maximize and minimize buttons, except you can customize the pictures in the buttons and attach custom code to clicks and double clicks.

Usage [*form.*]*vsInForm*.**ButtonsLeft** [= *setting%*]

Remarks The number of buttons ranges from 0 to 3 on each side.

These properties are used only if the CustomFrame property is set to True.

Default Value Zero

Data Type Integer

CapAlign Property

Description This property sets and returns the alignment of the text in the InForm parents caption bar.

Usage [*form.*]*vsInForm*.**CapAlign** [= *setting%*]

Remarks Valid settings 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

This property is used only if the CustomFrame property is set to True.

Default Value 4 - Center Center

Data Type Integer

CapColor, CapColorInactive Properties

Description These are the colors used to paint the text in the caption bar of the InForms parent. At run time, InForm chooses whether to use CapColor or CapColorInactive automatically, based on whether the InForms parent is active or not.

Usage [*form.*]*vsInForm*.**CapColor** [= *colorref&*]

Remarks These properties are used only if the CustomFrame property is set to True.

Setting either of these properties to zero causes the system default colors to be used. If you want the caption to be black, set these properties to one (if you simply pick black from the color box, VB will pick color zero and the control will use the default system colors).

Default Value Zero (use system colors)

Data Type Color (Long)

CapMultiLine Property

Description This property determines whether the text in the InForm parents caption bar should wrap, allowing for multi-line captions.

Usage [*form.*]*vsInForm*.**CapMultiLine** [= {**True|False**}]

Remarks This property is used only if the CustomFrame property is set to True.

Default Value False

Data Type Boolean

CapStyle Property

Description This property determines the appearance of the text in the caption bar of the InForms parent.

Usage [*form.*]*vsInForm*.**CapStyle** [= *setting%*]

Remarks Valid settings for this property are:
0 - Flat
1 - Raised
2 - Inset
3 - Raised Light
4 - Inset Light

For best effects with 3D captions, set the CapColor and CapColorInactive properties to dark gray.

This property is used only if the CustomFrame property is set to True.

Default Value 0 - Flat

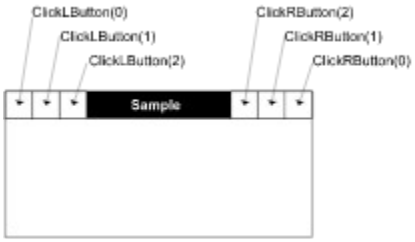
Data Type Integer

ClickLButton, ClickRButton Events

Description Fired after one of the custom caption buttons is clicked.

Syntax **Sub** *vsInForm*_**ClickLButton (Button as Integer)**

Remarks Buttons are numbered from the outside towards the center of the form, as shown in the diagram below:



ClipMon Property

Description Determines whether the InForm should monitor Clipboard activity.

Usage [*form.*]*vsInForm*.**ClipMon** [= {**True**|**False**}]

Remarks If this property is set to True, the InForm fires the NewClipboardData event whenever the contents of the Clipboard change.

Default Value False

Data Type Boolean

CustomFrame Property

Description Determines whether the InForm should take over painting and managing the non-client area of its parent.

Usage [*form.*]*vsInForm*.**CustomFrame** [= {**True**|**False**}]

Remarks If this property is set to true, the appearance of the InForms parent and its caption are controlled through the other InForm properties.

Default Value False

Data Type Boolean

DblClickCaption Event

Description Fired after the caption of the InForms parent is double-clicked.

Syntax **Sub** *vsInForm_***DblClickCaption ()**

Remarks Normally, you should trap this event to implement the Windows standard behavior: flip the window state between maximized and normal.

DblClickLButton, DblClickRButton Events

Description Fired after one of the custom caption buttons is double-clicked.

Syntax **Sub** *vsInForm_***DblClickLButton (Button as Integer)**

Remarks Buttons are numbered from the outside towards the center of the form. See diagram in the description of the ClickLButton event.

DropFile Event

Description Fired when the user drops one or more files from another application such as the Program Manager or the File Manager into the InForms parent.

Syntax **Sub** *vsInForm_***DropFile ()**

Remarks To enable this feature, you must set the InForms AcceptFiles property to True.

To find out which files were dropped into the control, use the InForms NumFiles and FileName properties.

Example

```
Sub vsInForm_DropFile ()
 Dim i%
 For i = 0 to vsInform.NumFiles
 ProcessFile vsInForm.FileName (i)
 Next
End Sub
```

FileName Property

Description This property returns the names of files dropped into the InForms parent.

Usage *filename$* = [*form.*]*vsInForm*.**FileName(***index%***)**

Remarks This property is read-only and should only be used while handling the InForms DropFile event. The number of files dropped can be retrieved by reading the InForms NumFiles property.

See the description of the DropFile event for an example.

Data Type String Array

FrameButtons Property

Description This property affects the look of the buttons on InForms custom frame. Is set to True, a thin black border is drawn around the buttons. If set to False, no border is drawn and the buttons appear immersed in the caption bar.

Usage [*form.*]*vsInForm*.**FrameButtons** [= {**True**|**False**}]

Remarks This property has no effect if the CustomFrame property is set to False or if the custom frame has no buttons.

Default Value True

Data Type Boolean

FrameColor, FrameColorInactive Properties

Description These are the colors used to paint the frame of the InForms parent. At run time, InForm chooses whether to use FrameColor or FrameColorInactive automatically, based on whether the InForms parent is active or not.

Usage [*form.*]*vsInForm*.**FrameColor** [= *colorref&*]

Remarks These properties are used only if the CustomFrame property is set to True.

Setting either of these properties to zero causes the system default colors to be used. If you want the frame to be black, set these properties to one (if you simply pick black from the color box, VB will pick color zero and the control will use the default system colors).

Default Value Zero (use system colors)

Data Type Color (Long)

FrameCorners Property

Description This property determines whether the InForm should mark the resizing corners on its parents frame.

Usage [*form.*]*vsInForm*.**FrameCorners** [= {**True**|**False**}]

Remarks This property is used only if the CustomFrame property is set to True.

This property only affects the window appearance, not its resizing behavior.

Default Value True

Data Type Boolean

FrameSizing Property

Description This property determines whether the InForm should allow its parent to be resized in both directions, in one direction, or not at all.

Usage [*form.*]*vsInForm*.**FrameSizing** [= *setting%*]

Remarks Valid settings for this property are:

0 - None	Move freely, but no resizing
1 - Horizontal	Move freely, resize width only
2 - Vertical	Move freely, resize height only
3 - Both	Move freely, resize freely
4 - Locked	No moving, no resizing

This property is used only if the CustomFrame property is set to True.

Default Value 3 - Both

Data Type Integer

FrameStyle Property

Description This property determines the appearance of the resizing frame around the InForms parent.

Usage [*form.*]*vsInForm*.**FrameStyle** [= *setting%*]

Remarks Valid settings for this property are:
0 - None
1 - Classic
2 - Outside
3 - Raised Form
4 - Raised Frame

This property is used only if the CustomFrame property is set to True.

Default Value 1 - Classic

Data Type Integer

FrameWidth Property

Description This property sets or returns the width of the frame around the InForms parent, in pixels.

Usage [*form.*]*vsInForm*.**FrameWidth** [= *width%*]

Remarks This property is used only if the CustomFrame property is set to True.

Setting this property to zero causes the system default frame width to be used. If you want a window with no frame, set the FrameStyle property to None.

Default Value Zero (use system value)

Data Type Integer

FreeGDI Property

Description This property returns the percentage of free Windows GDI resources.

Usage *GDI%* = [*form.*]*vsInForm*.**FreeGDI**

Remarks This property is read-only.

Data Type Integer

FreeMemory Property

Description This property returns the amount of free memory available to Windows, in bytes.

Usage *Mem&* = [*form*.]*vsInForm*.**FreeMemory**

Remarks This property is read-only.

Data Type Long

FreeSystem Property

Description This property returns the percentage of free Windows system resources.

Usage *Sys%* = [*form.*]*vsInForm*.**FreeSystem**

Remarks This property is read-only.

Data Type Integer

FreeUser Property

Description This property returns the percentage of free Windows user resources.

Usage *User%* = [*form.*]*vsInForm*.**FreeUser**

Remarks This property is read-only.

Data Type Integer

MaxHeight, MinHeight Properties

Description These properties allow you to set limits, in Twips, to the height of the InForms parent.

Usage [*form.*]*vsInForm*.**MaxHeight** [= *maxhei&*]

Remarks These properties are useful for windows that only show information along one direction, such as roll-up toolbars or VBs main window.

These properties are used only if the CustomFrame property is set to True.

Default Value Zero

Data Type Long

MaxWidth, MinWidth Properties

Description These properties allow you to set limits, in Twips, to the width of the InForms parent.

Usage [*form.*]*vsInForm*.**MaxWidth** [= *maxwid&*]

Remarks These properties are useful for windows that only show information along one direction, such as roll-up toolbars or VBs main window.

These properties are used only if the CustomFrame property is set to True.

Default Value Zero

Data Type Long

Move Event

Description Fired after the InForms parent has been moved.

Syntax **Sub** *vsInForm_***Move ()**

Remarks To enable this feature, you must set the InForms ClipMon property to True.

NewClipboardData Event

Description Fired whenever new data is copied to the Windows Clipboard. You can then use VBs Clipboard object to examine, retrieve, or change the clipboard contents.

Syntax **Sub** *vsInForm_***NewClipboardData ()**

Remarks To enable this feature, you must set the InForms ClipMon property to True.

NumFiles Property

Description This property returns the number of files dropped into the InForms parent.

Usage *numfiles%* = [*form.*]*vsInForm*.**NumFiles**

Remarks This property is read-only and should only be used while handling the InForms DropFile event. The names of the files dropped can be retrieved by reading the InForms FileName property.

See the description of the DropFile event for an example.

Data Type Integer

OnTop Property

Description Setting this property to True forces the InForms parent to remain on top of other windows, even when it is not active.

Usage [*form.*]*vsInForm*.**OnTop** [= {**True**|**False**}]

Default Value False

Remarks This is especially useful for floating toolbars and other small windows. A good way to implement OnTop functionality is to use one of the custom caption buttons to turn it on and off (see the example on page 6).

Data Type Boolean

PictLeft*, PictRight* Properties

Description These properties are used to select the pictures for the InForm parents custom caption bar buttons. There are 6 properties in total, left and right ranging from 0 to 2.

Usage [*form.*]*vsInForm*.**PictLeft0** [= *ImageControl*]

Remarks The InForm draws the bevels automatically. All you have to supply is a small bitmap to be drawn on the button. For best results, make sure the background of the bitmaps is light gray, so it blends with the button background.

These properties could have been implemented as picture arrays, but VB does not allow setting array properties at design-time.

Data Type Picture

Resize Event

Description Fired after the InForms parent has been resized.

Syntax **Sub** *vsInForm_***Move ()**

Remarks There is no need for this event if the InForms parent is a form, since forms have their own Resize event. However, you need this event to trap resizing of other controls.

To enable this feature, you must set the InForms ClipMon property to True.

Version Property

Description This property returns the version of the InForm control currently loaded in memory.

Usage *CheckVer%* = [*form.*]*vsInForm*.**Version**

Remarks You may want to check this value at the Form_Load event, to make sure the version that is executing is at least as current as the version used to develop your application.

The version number is a three digit integer where the first digit represents the major version number and the last two represent the minor version number. For example, version 3.5 would return 350.

This property is read-only.

Data Type Integer

Printer Reference

Description The VideoSoft Printer control allows you to create printed output quickly and easily. It supports multiple columns, text wrapping, headers and footers, and pictures. Best of all, it lets you do Print Previews just by setting a single property.

File Name VSVIEW.VBX

Object Type vsPrinter

Remarks The Printer control has a fixed coordinate system. It uses Twips as units, with the origin located at the top left of the page. To specify measurements in inches or points, remember that one inch is equivalent to 1440 Twips, and that one point is equivalent to 20 Twips.

For convenience, two measurements are expressed in units other than Twips. Font sizes are expressed in points, and line spacing is expressed specified as a percentage of the current font height.

When used in print preview mode, the scale is adjusted automatically so that a whole page always fits the control. This makes it easy to implement zooming in the print preview: all you have to do is resize the Print control, and its contents will stretch or shrink as needed.

Printer Summary

Properties (default: Paragraph)

(Abort)	* AbortCaption	* AbortWindow
* Action	BackColor	BorderStyle
* BrushColor	* BrushStyle	* Columns
* CurrentColumn	* CurrentLine	* CurrentPage
* CurrentX	* CurrentY	* Device
* Devices	* DPI	* Draw
* Driver	Enabled	* Error
* FileName	FontBold	FontItalic
FontName	FontSize	FontUnderline
* Footer	* Hdc	* HdrColor
* HdrFontBold	* HdrFontItalic	* HdrFontName
* HdrFontSize	* HdrFontUnderline	* Header
Height	HelpContextID	* IndentFirst
* IndentLeft	* IndentRight	* IndentTab
Index	Left	* LineSpacing
* MarginBottom	* MarginLeft	* MarginRight
* MarginTop	* Measure	MousePointer
Name	* NDevices	* NPorts
* Orientation	* PageBorder	* PageHeight
* PageWidth	* Paragraph	Parent
* PenColor	* PenStyle	* PenWidth
* PhysicalPage	* Picture	* Polygon
* PolyLine	* Port	* Ports
* Preview	* PreviewMode	* PreviewPage
* SpaceAfter	* Table	* TableBorder
* TableSep	Tag	* Text
* TextAngle	* TextAlign	* TextColor
* TextHei	* TextWid	Top
* Version	Visible	Width
* X1	* X2	* Y1
* Y2		

Events

Click	DblClick	DragDrop
DragOver	* EndDoc	* EndPage
* Error	KeyDown	KeyPress
KeyUp	MouseDown	MouseMove
MouseUp	* NewColumn	* NewLine
* NewPage	* NewTableCell	* StartDoc

AbortCaption Property

Description Sets the caption for the default Abort Window.

Usage [*form.*]*vsPrinter*.**AbortCaption** [= *caption$*]

Remarks See the AbortWindow property for details on the default Abort Window.

Default Value Printing...

Data Type String

AbortWindow Property

Description Determines whether the Printer should automatically display an Abort Window while it prints.

Usage [*form.*]*vsPrinter*.**AbortWindow** [= {**True**|**False**}]

Remarks The automatic Abort Window shows the name of the document being printed (from the FileName property), the output device, port, and the current page. If the user presses the Abort button, printing stops and an Error event is fired.

You may set the caption of the Abort Window with the AbortCaption property.

Set this property to False if you want to display a custom Abort Window.

Default Value True

Data Type Boolean

Action Property

Description This property allows you to control the printer.

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

Remarks Valid settings for this property are:
0 - None
1 - Print File
2 - Choose Printer & Print File
3 - StartDoc
4 - NewPage
5 - NewCol
6 - EndDoc
7 - AbortDoc
8 - Print Page
9 - Choose Printer & Print Page
10 - Copy Page
11 - Print All Pages
12 - Choose Printer & Print All Pages

The *Print File* action prints an entire text file using the current settings for font, margins, header and footer, and number of columns. The file name must be specified in advance by setting the FileName property. For example:

```
vsPrinter.FileName = C:\AUTOEXEC.BAT
vsPrinter.Action = 1      print the AUTOEXEC.BAT file
```

The *Choose Printer & Print File* action is similar to the *Print File* action, except it displays a dialog box that allows the user to choose the printer to be used.

The *StartDoc* and *EndDoc* properties are used to create and end a print job. All other printing commands should be called between these actions. The example below illustrates this:

```
vsPrinter.Action = 3              start document
  If vsPrinter.Error Then         always check for errors
    Exit Sub
  EndIf
  vsPrinter = VideoSoft         print some text
  vsPrinter =  2625 Alcatraz Avenue, Suite 271
  vsPrinter =  Berkeley, CA 94705
  vsPrinter =  (510) 547-7295 (phone)
  vsPrinter =  (510) 547-1084 (fax)
vsPrinter.Action = 6              end document
```

The *NewPage* and *NewCol* actions can be used to force page and column breaks. If you use *NewCol* at the last column on a page, the current page is ejected and a fresh one is started.

The *AbortDoc* action cancels the current print job. This action should be used when you provide your own Abort window.

The *Print Page* action prints the current preview page to the current printer. The *Choose Printer & Print Page* action shows a printer selection dialog box, then prints the current preview page to the selected printer. If nothing has been printed to the current preview page, this action does nothing.

The *Copy Page* action copies the current page to the clipboard. If nothing has been printed to the current preview page, this action does nothing.

The *Print All Pages* and *Choose Printer & Print All Pages* actions send all preview pages directly to the printer, so you dont have to call your printing routine twice. This is normally faster than printing the normal way, but the quality of metafile pictures will degrade when printed directly fro mthe preview pages.Naturally, these actions only work if there preview pages available.

The Action property only acts at run-time, but the above list is also shown at design-time, on the properties window. This makes it easy for you to remember the code for each action.

This property is read-only.

Data Type Integer

BrushColor Property

Description Sets or returns the color to be used when filling objects.

Usage [*form.*]*vsPrinter*.**BrushColor** [= *colorref&*]

Remarks This property affects the drawing of rectangles and ellipses (see the Draw property) as well as polygons (see the Polygon property).

Default Value Black

Data Type Long (Color)

BrushStyle Property

Description Sets or returns the style to be used when filling objects.

Usage [*form.*]*vsPrinter*.**BrushStyle** [= *colorref&*]

Remarks Valid settings for this property are:
0 - Solid
1 - Transparent
2 - Horizontal Line
3 - Vertical Line
4 - Upward Diagonal
5 - Downward Diagonal
6 - Cross
7 - Diagonal Cross

This property affects the drawing of rectangles and ellipses (see the Draw property) as well as polygons (see the Polygon property).

Default Value 0 - Solid

Data Type Long (Color)

Columns Property

Description Sets or returns the number of columns to be used when printing text.

Usage [*form.*]*vsPrinter*.**Columns** [= *numcolumns%*]

Remarks The setting of the Columns property affects the flow of text as you send it to the printer. The Print control does word wrapping, line, column, and page feeds automatically for you.

Default Value 1

Data Type Integer

CurrentColumn, CurrentLine, CurrentPage Properties

Description These properties return the current page, column and line being printed by the Printer control.

Usage *page%* = [*form.*]*vsPrinter*.**CurrentPage**

Remarks These properties are read-only.

Data Type Integer

CurrentX, CurrentY Properties

Description These properties are similar to the standard CurrentX and CurrentY properties available in VBs Picture control and Printer object.

Usage [*form.*]*vsPrinter*.**CurrentX** [= *newpos&*]

Remarks As text is printed, these properties change to reflect the new cursor position. You may modify these properties directly if you want to set the cursor at a specific position. This may be necessary, for example, when centering text or printing tables.

Data Type Long (Twips)

DPI Property

Description This property returns the resolution of the current printer, expressed as dots per inch.

Usage *printres%* = [*form.*]*vsPrinter*.**DPI**

Remarks This property is read-only.

Data Type Integer

Device Property

Description This property sets or returns the name of the default Windows printer.

Usage [*form.*]*vsPrinter*.**Device** [= *devname$*]

Remarks You can use this property to provide user feedback so they know which printer is being used or to set the default Windows printer.

You cannot change this property while a job is printing. Trying to do so will trigger an Error event.

You can only set this property to a valid device name. To obtain a list of valid device names, read the Devices array property. Trying to set this property to an invalid device name will trigger an Error event.

Changing this property will affect all Windows-based applications. Therefore, you may want to save the current printer before changing it and restore it later, as shown in the example below:

```
 ** this example lists all printing devices installed

 save current device and port to restore later
dev$ = vsPrinter.Device
prt$ = vsPrinter.Port

 loop through devices
For i = 0 to vsPrinter.NDevices - 1
  vsPrinter.Device = vsPrinter.Devices(i)

   print all ports this device is connected to
  For j = 0 to vsPrinter.NPorts
    Debug.Print vsPrinter.Device;   on ; vsPrinter.Ports(j)
  Next
Next

 restore original settings
vsPrinter.Device = dev$
vsPrinter.Port = prt$
```

Changing this property automatically updates the Driver, Port, and DPI properties.

Data Type String

Devices Property

Description This property array contains a list of the printing devices available.

Usage *devname$* = [*form*.]*vsPrinter*.**Devices(***devindex%***)**

Remarks To find out how many devices are available, read the NDevices property.

For an example, see the description of the Device property.

This property is read only.

Data Type String Array

Draw Property

Description This property allows you to draw lines, rectangles, and ellipses on the printer.

Usage [*form.*]*vsPrinter*.**Draw** = *object%*

Remarks Valid settings for this property are:
0 - Nothing
1 - Line
2 - Rectangle
3 - Ellipse

All objects are drawn extending between the (x1, y2) and (x2, y2) points defined by the X1, Y1, X2, and Y2 properties.

Objects are drawn with the current pen and filled with the current brush. Pen and brush attributes are defined with the Pen* and Brush* properties.

This property only acts at run-time, but the above list is also shown at design-time, on the properties window. This makes it easy for you to remember the code for each object.

This property is write-only.

Data Type Integer

Driver Property

Description This property returns the name of the current printer driver.

Usage *drivername$* = [*form.*]*vsPrinter*.**Driver**

Remarks This property is read-only.

Data Type String

EndDoc Event

Description This event gets fired after a document finishes printing.

Syntax **Sub** *vsPrinter*_**EndDoc ()**

EndPage Event

Description This event is fired right before each page is ejected.

Syntax **Sub** *vsPrinter*_**EndPage ()**

Error Event

Description This event is fired when an error is detected while printing. The type of error can be determined by checking the Error property.

Syntax **Sub** *vsPrinter_***Error ()**

Error Property

Description This property is set whenever an error is detected while printing, or when the user aborts the printing process.

Usage *errcode%* = [*form.*]*vsPrinter*.**Error**

Remarks You should always check this property after starting a new print job and also while printing. The error code is reset whenever a new document is started.

The error codes are:
0 - No Error
1 - Cant Open File
2 - Line Too Long
3 - Cant Access Printer
4 - Cant Start Job
5 - User Aborted
6 - Already Printing

The *Cant Open File* error occurs when you try to print a file using Action = 1 and the file specified by the FileName property cannot be opened.

The *Line Too Long* error occurs when a text file is being printed but a line is too long to fit in memory. The line gets truncated.

The *Cant Access Printer* and *Cant Start Job* errors occur when the printer is not available.

The *User Aborted* error occurs when the users clicks the Abort button while a document is printing.

The *Already Printing* error occurs when you try to start a new document while a document is being printed.

This property is read-only

Default Value 0 - No Error

Data Type Integer

FileName Property

Description This property sets or retrieves the name of the text file being printed with the Action = 1 action. The file must exist, or an Error event is fired.

Usage [*form.*]*vsPrinter*.**FileName** [= *filename$*]

Remarks This property also sets the name of the current job, a string that appears in the default Abort Window and on the Print Manager list.

Default Value (Empty String)

Data Type String

Footer Property

Description This property sets or retrieves the text of the footer printed at the bottom of every page.

Usage [*form.*]*vsPrinter*.**Footer** [= *footer$*]

Remarks The footer is composed of three sections, separated by pipe characters (|). The first section is left-justified, the second is centered, and the third is right-justified.

You may also include a page number field by embedding a %d code in the string. Do not use any percent signs in footers except for the page code.

For example, the following footer would print the file name and page number on the left and right corners of every page:

vsPrinter.Footer = vsPrinter.FileName + ||Page %d

The footer is printed using the font defined by the HdrFont* properties.

Default Value (Empty String)

Data Type String

Hdc Property

Description This property returns the Windows hDC (handle to device context) being used by the printer control.

Usage *hdc%* = [*form.*]*vsPrinter*.**Hdc**

Remarks This property is useful if you wish to call Windows API functions to draw to the printer.

The value of the Hdc property corresponds to a printer or metafile hDC, depending on whether the control is drawing to the printer or to the screen (in print preview mode).

If theres no open document, the Hdc property returns zero.

This property is read-only.

Data Type Integer

HdrColor Property

Description This property sets or retrieves the color of the font used to draw the document headers and footers.

Usage [*form.*]*vsPrinter*.**HdrColor** [= *colorref&*]

Default Value 0 (Black)

Data Type Long (Color)

HdrFont* Properties

Description These properties are similar to the default Font* properties, except they define the font used for printing the headers and footers, while the standard properties define the font used for printing regular text.

Usage

[*form.*]*vsPrinter*.**HdrFontName** [= *fontname$*]
[*form.*]*vsPrinter*.**HdrFontSize** [= *size%*]
[*form.*]*vsPrinter*.**HdrFontBold** [= {**True**|**False**}]
[*form.*]*vsPrinter*.**HdrFontItalic** [= {**True**|**False**}]
[*form.*]*vsPrinter*.**HdrFontUnderline** [= {**True**|**False**}]

Header Property

Description This property sets or retrieves the text of the header printed at the top of every page.

Usage [*form.*]*vsPrinter*.**Header** [= *header$*]

Remarks The header is composed of three sections, separated by pipe characters (|). The first section is left-justified, the second is centered, and the third is right-justified.

You may also include a page number field by embedding a %d code in the string. Do not use any percent signs in headers except for the page code.

For example, the following header would print the file name and page number on the top left and right corners of every page:

vsPrinter.Header = vsPrinter.FileName + ||Page %d

The header is printed using the font defined by the HdrFont* properties.

Default Value (Empty String)

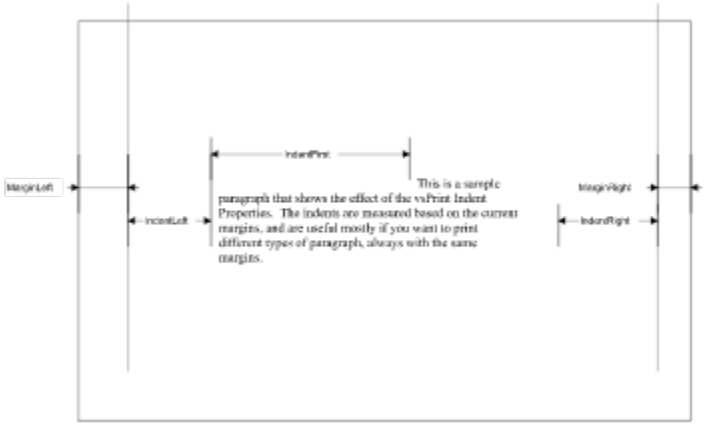
Data Type String

IndentLeft, IndentRight, IndentFirst Properties

Description These properties set or retrieve the indentation to be used when printing paragraphs.

Usage [*form.*]*vsPrinter*.**IndentLeft** [= *indent&*]

Remarks The indents are measured from the current margins, as illustrated below:



Default Value 0

Data Type Long (Twips)

IndentTab Property

Description This property sets or retrieves the size of the tab stops used for printing text.

Usage [*form.*]*vsPrinter*.**IndentTab** [= *indenttab&*]

Remarks Whenever a tab character is detected in the text, the Print control advances the print position to the next multiple of the IndentTab. If the next position is beyond the end of the current line, then the output advances to the next line.

Default Value 770 Twips (1/2)

Data Type Long (Twips)

LineSpacing Property

Description This property sets or retrieves the amount of space to leave between lines of text.

Usage [*form.*]*vsPrinter*.**LineSpacing** [= *linespacing%*]

Remarks Line spacing is expressed as a percentage of the current font height.

The following table summarizes common settings for the LineSpacing property:

Setting	Effect
100%	Single line spacing
150%	1.5 line spacing
200%	Double line spacing
50%	Half line spacing.

Default Value 100 (single line spacing)

Data Type Integer

MarginBottom, MarginTop, MarginLeft, MarginRight Properties

Description These properties set or retrieve the distance between the edge of the page and the printed text.

Usage [*form.*]*vsPrinter*.**MarginBottom** [= *margin&*]

Default Value 770 Twips (1/2)

Data Type Long (Twips)

Measure Property

Description This property is used to measure the width and height of a string printed with the current font on the current device.

Usage [*form.*]*vsPrinter*.**Measure** [= *text$*]

Remarks To measure the text, assign it to this property. The width and height of the text are returned, in the TextWid and TextHei properties.

This property is write-only.

Data Type String

NDevices Property

Description This property returns the number of printing devices currently installed.

Usage *devices%* = [*form.*]*vsPrinter*.**NDevices**

Remarks To acess the name of a specific device, use the Devices() string array.

This property is read-only.

Data Type Integer

NewColumn Event

Description This event is fired after each column break.

Syntax **Sub** *vsPrinter*_**NewColumn** ()

NewLine Event

Description This event is fired after each line break.

Syntax **Sub** *vsPrinter*_**NewLine** ()

NewPage Event

Description This event is fired after each page is ejected.

Syntax **Sub** *vsPrinter_***NewPage ()**

Remarks When this event is fired, the page is still blank. You can use this event to print custom headers and footers. You can even create watermarks, gray or transparent text and graphics that appear behind the document text.

NewTableCell Event

Description While printing a table with the Table property, this event is fired before each cell is printed.

Syntax **Sub** *vsPrinter_***NewTableCell (Row As Integer, Column As Integer, Cell As String)**

Remarks This event allows you to customize the format of individual table cells.

All parameters are read-only. The Row and Column parameters identify the cell about to be printed, and the Cell parameter contains the text that will be printed.

For example, the following code customizes the third column of a table. It prints negative numbers in bold red and positive numbers in bold green:

```
Sub VSPrint_NewTableCell (Row%, Column%, Cell$)
    ** were only customizing column 3:
  If Column = 3 Then
    VSPrinter1.FontBold = True
    If Val (Cell$) < 0 Then
      VSPrinter1.TextColor = RGB (255, 0, 0)
    Else
      VSPrinter1.TextColor = RGB (0, 255, 0)
   ** restore regular formatting
  Else
    VSPrinter1.FontBold = False
    VSPrinter1.TextColor = 0
  End If
End Sub
```

NPorts Property

Description This property returns the number of ports to which the default Windows printer is connected.

Usage *nports%* = [*form.*]*vsPrinter*.**NPorts**

Remarks This property will return 1 unless you have identical printers connected to different ports. For example, if the default Windows printer is an HP LaserJet IV connected to LPT1: and theres another HP LaserJet IV connected to LPT2:, then NPorts would return 2, and the Ports property array would contain the strings LPT1: and LPT2:.

For an example, see the description of the Device property.

This property is read-only.

Data Type Integer

Orientation Property

Description This property sets or retrieves the current paper orientation setting.

Usage [*form.*]*vsPrinter*.**Orientation** [= *orient%*]

Remarks Valid settings for this property are:
0 - Portrait
1 - Landscape

Most popular printers support portrait and landscape orientations, but some do not. Therefore, after changing the paper orientation through code, you may want to check and make sure the change actually took place. You can do this simply by reading back the Orientation value, as show below:

```
vsPrinter.Orientation = 1                    ** try landscape
If vsPrinter.Orientation <> 1 Then           ** check if it worked
   MsgBox Sorry, unable to switch to landscape
End If
```

Data Type Integer

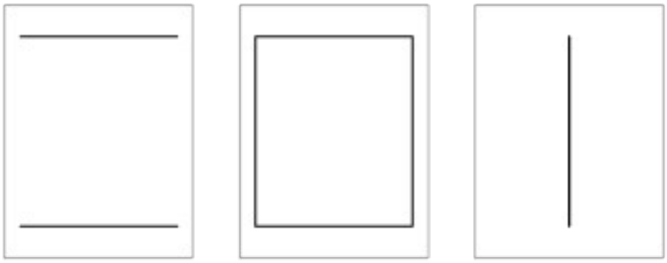
PageBorder Property

Description This property is used to set or retrieve the type of border to be drawn around each page.

Usage [*form.*]*vsPrinter*.**PageBorder** [= *setting%*]

Remarks Valid settings for this property are:
0 - None
1 - Bottom
2 - Top
3 - Top & Bottom
4 - Box
5 - Columns
6 - Columns, Top, & Bottom
7 - All

The effect of these settings is illustrated below:



The border is drawn using the pen defined by the Pen* properties.
See also the Table property.

Default Value 3 - Top & Bottom

Data Type Integer

PageHeight, PageWidth Properties

Description These properties return the size of a page to be printed on the current printer, in Twips.

Usage *pagehei%* = [*form.*]*vsPrinter*.**PageHeight**

Remarks These values are useful when drawing graphics on the page and also when implementing print preview. For example, to preview a printed page at 60% of its actual size, one could use the following code:

```
vsPrinter.Width = vsPrinter.PageWidth * 0.6
vsPrinter.Height = vsPrinter.PageHeight * 0.6
```

The above code would resize the Printer control to 60% of the size of a printed page, along with its contents.

Data Type Long (Twips)

Paragraph Property

Description This property is used to print a paragraph. The Printer control takes care of all the justification, word wrapping, and text flow.

Usage [*form.*]*vsPrinter*[.**Paragraph**] = *text$*

Remarks After printing a string with this property, the Printer control will automatically skip a line and get ready for the next paragraph. If you want to print a paragraph in pieces, use the Text property instead.

This property is write-only.

Data Type String

PenColor Property

Description This property sets or returns the color of the pen used to draw all graphics and page borders.

Usage [*form.*]*vsPrinter*.**PenColor** [= *colorref&*]

Remarks This property does not affect printed text. To change text color, use the TextColor and HdrColor properties.

Default Value 0 (Black)

Data Type Long (Color)

PenStyle Property

Description This property sets or returns the style of the pen used to draw all graphics and page borders.

Usage [*form.*]*vsPrinter*.**PenStyle** [= *setting%*]

Remarks Valid settings for this property are:
0 - Solid
1 - Dash
2 - Dot
3 - Dash-Dot
4 - Dash-Dot-Dot
5 - Transparent
6 - Inside Solid

Note that non-solid styles only work when the PenWidth property is set to zero. This is a GDI limitation.

Default Value 0 -Solid

Data Type Integer

PenWidth Property

Description This property sets or returns the width of the pen used to draw all graphics and page borders.

Usage [*form.*]*vsPrinter*.**PenWidth** [= *width%*]

Remarks Setting PenWidth to zero causes the thinnest possible pen to be used. The pen width units are Twips.

Default Value 0

Data Type Integer

PhysicalPage Property

Description This property determines whether the logical page used by the Printer control should correspond to the entire physical page or only to its printable area.

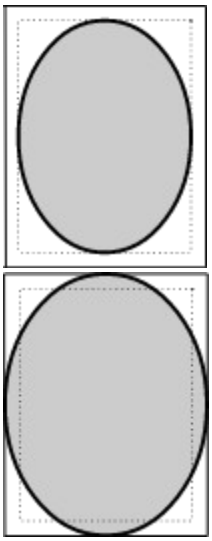
Usage [*form.*]*vsPrinter*.**PhysicalPage** [= {**True | False**}]

Remarks Most printers have a "logical" paper size that corresponds to the printer's printable area and a "physical" paper size that corresponds to the actual page size. The physical paper size is always a little larger than the logical paper size.

If the PhysicalPage property is set to True, the vsPrinter control draws and prints to the physical page. If the PhysicalPage property is set to False, the Printer control draws and prints to the logical page. This affects the page dimensions returned by the PageWidth and PageHeight properties, and it also offsets the origin of the vsPrinter's coordinate system.

The pictures below show the effect of the PhysicalPage property when the following code is executed. Note how the edges of the oval are not printed when PhysicalPage is set to True.

```
vsPrinter.Action = 3
vsPrinter.X1 = 0
vsPrinter.Y1 = 0
vsPrinter.X2 = vsPrinter.PageWidth
vsPrinter.Y2 = vsPrinter.PageHeight
vsPrinter.Draw = 3    oval
vsPrinter.Action = 6
```



vsPrinter.PhysicalPage = False vsPrinter.PhysicalPage = True

Normally, this property should be set to False, to ensure that all output sent to the printer will actually be reproduced on the page, and not truncated if it's too close to the edges of the page.

In a few special situations, however, you may want to set this property to True, so you can print text and graphics at exact positions with respect to the physical page. This is mainly useful for printing labels and for filling pre-printed forms.

Default Value False

Data Type Boolean

Picture Property

Description This property allows you to print pictures on a page, at a specific position.

Usage [*form.*]*vsPrinter*.**Picture** = *Image.Picture*

Remarks The position of the picture is determined by the X1, Y1, X2, and Y2 properties.

The Printer control can handle bitmaps, icons, and metafiles in print and preview mode.

Note that previewing large pictures requires a lot of memory. If you are going to print long documents with lots of large pictures, you may want to provide a draft preview mode which displays picture holders instead of actual pictures, as shown in the code below:

```
vsPrinter.Action = 3                        start document
vsPrinter.x1 = vsPrinter.MarginLeft + 1440   1 inch from left
vsPrinter.y1 = vsPrinter.MarginTop + 1440    1 inch from top
vsPrinter.x2 = vsPrinter.x1 + 2880          2 inches wide
vsPrinter.y2 = vsPrinter.y1 + 2880          2 inches tall
If DraftPreview Then
  vsPrinter.Draw = 2                          picture holder
Else
  vsPrinter.Picture = Picture1                picture
End If
vsPrinter.Action = 6                        end document
```

This property is write-only.

Data Type Picture

PolyLine Property

Description This property allows you to plot a line composed of many segments.

Usage [*form.*]*vsPrinter*.**PolyLine** = *points$*

Remarks The *points$* string contains a sequence of X, Y coordinates, in Twips, separated by spaces. Only the integer part of the coordinates is used. For example, the following code would draw a sine wave one inch tall across the page:

```
s =
For i = 0 to 12 * PI Step PI/4
   x = vsPrinter.PageWidth * i / (12 * PI)
   y = vsPrinter.PageHeight / 2 + 770 * Sin(i)
   s = s + Str(Int(x)) + Str(Int(y))
Next
```

You could create complex lines by repeatedly setting the Draw property to *Line* and updating X1, Y1, X2, and Y2, but the PolyLine is usually a more convenient and efficient way of obtaining the same results.

The line is drawn with the current pen, defined by the Pen* properties.

This property is write-only

Data Type String

Polygon Property

Description This property allows you to plot an arbitrary filled polygon. It is similar to the PolyLine property except that it produces a closed figure.

Usage [*form.*]*vsPrinter*.**Polygon** = *points$*

Remarks For details on the syntax of the *points$* parameter, see the description of the PolyLine property.

Data Type String

Port Property

Description This property returns the name of the port to which the current printer is connected.

Usage *portname$* = [*form.*]*vsPrinter*.**Port**

Remarks This property is read-only.

Data Type String

Ports Property

Description This property array contains a list of the ports to which the default Windows printer is connected.

Usage *portname$* = [*form.*]*vsPrinter*.**Ports(***portindex%***)**

Remarks To find out how many ports are connected to the default Windows printer, read the NPorts property.

For an example of how to use this property, see the description of the Device property.

This property is read only.

Data Type String Array

Preview Property

Description This property determines whether Print control output should be sent to the printer or to the screen.

Usage [*form.*]*vsPrinter*.**Preview** [= {**True**|**False**}]

Remarks If the Preview property is set to False, the Print control sends its output to the printer. Unless the AbortWindow property is set to False, a default Abort window is created and shown to allow the user to abort the print job.

If the Preview property is set to True, the Print control saves all output in memory and allows you to display it, one page at a time, on the Print control itself. You can then switch pages by changing the PreviewPage property and scale the output by changing the size of the Print control.

This allows you to use the same code to create printed output and to perform print previews. The code below illustrates how this can be done:

```
 This routine sends output to the printer
Sub PrintButton_Click ()
  vsPrinter.Preview = False
  DoPrinting
End Sub

 This routines sends output to the Printer control
Sub Preview_Click ()
  vsPrinter.Preview = True
  vsPrinter.PreviewPage = 1
  DoPrinting
End Sub

 This routine switches preview page
Sub PreviewPage_Click (Index as Integer)
  Select Case Index
    Case 0:
      vsPrinter.PreviewPage = 1                                  first
    Case 1:
      vsPrinter.PreviewPage = vsPrinter.PreviewPage + 1  next
    Case 2:
      vsPrinter.PreviewPage = vsPrinter.PreviewPage - 1  previous
    Case 3:
      vsPrinter.PreviewPage = vsPrinter.CurrentPage         last
  End Select
End Sub
```

In preview mode, the Printer control stretches the page image to fill the control. This makes it easy to implement print preview zooming, as illustrated by the code below:

```
 This routine zooms the preview page
Sub ZoomPreviewPage (ZoomFactor as Integer)
  vsPrinter.Width = vsPrinter.PageWidth * ZomFactor / 100
  vsPrinter.Height = vsPrinter.PageHeight * ZomFactor / 100
End Sub
```

The maximum number of pages in a preview document is 100.

Default Value False

Data Type Boolean

PreviewMode Property

Description This property allows you to make the colors of the preview image match the screen, printer, or be always black and white.

Usage [*form.*]*vsPrinter*.**PreviewMode** [= *mode%*]

Remarks Valid settings for this property are:
0 - Screen Compatible (usually color)
1 - Printer Compatible (usually black and white)
2 - Force Monochrome (always black and white)

This property is useful when you are going to print documents that consist mostly of black and white text. In this case, you can save a lot of memory and speed up program execution by setting the PreviewMode property to *2 - Force Monochrome.*

Default Value 0 - Screen Compatible

Data Type Integer

PreviewPage Property

Description This property sets or retrieves the number of the page being previewed.

Usage [*form.*]*vsPrinter*.**PreviewPage** [= *page%*]

Remarks Valid page numbers start at one and go up to the last page printed. The number of the last page can be retrieved by reading the CurrentPage property.

Default Value 1

Data Type Integer

SpaceAfter Property

Description This property sets or retrieves the amount of vertical spacing between consecutive paragraphs. Units are Twips.

Usage [*form.*]*vsPrinter*.**SpaceAfter** [= *space&*]

Remarks This space is automatically added after each paragraph when you use the Paragraph property to print text.

Default Value 0

Data Type Long (Twips)

StartDoc Event

Description This event gets fired before a document starts printing.

Syntax **Sub** *vsPrinter_***StartDoc ()**

Table Property

Description This property is used to print tables. The Printer control takes care of all the justification, word wrapping, and text flow.

Usage [*form.*]*vsPrinter*.**Table** = *table$*

Remarks The table string is divided into rows and columns by special characters defined through the TableSep property. By default, rows are separated by semi-colons (;) and columns by column pipes (|), as shown in the example below.

The first row contains formatting information only and is not printed. The formatting information consists of column alignment and width. The alignment is specified through the special characters <, >, and ^, for left, right, and center. The default is left alignment. The column widths are specified in Twips.

The TableBorder and Pen* properties determine the appearance of lines between the table entries, and you can customize the appearance of individual table cells by trapping the NewTableCell event.

For example, the following code draws a table with four rows and four columns. The columns are 0.5, 2, 1, and 2 inches wide:

```
 ** define format: first field is right-justified
fmt = >770|2880|1440|2880;
 ** print table header in bold
tbl = fmt + Code|Name|Phone|Address;
vsPrinter.FontBold = True
vsPrinter.Table = tbl
 ** print table body in normal
vsPrinter.FontBold = False
tbl = fmt + 00123|John|415/324-2323|2312 Embarcadero South;
tbl = tbl + 00432|Richard|303/321-2312|3212 Market;
tbl = tbl + 02312|Susan|415/342-3421|666 Lombard;
vsPrinter.Table = tbl
```

Before each table cell is printed, the vsPrinter control fires the NewTableCell event. You may trap this event to customize the appearance of individual cells.

Tables are printed at the current vertical position on the page, specified by the CurrentY property. Tables are aligned on the page like paragraphs: left, center or right, depending on the setting of the TextAlign property.

This property is write-only.

Data Type String

TableBorder Property

Description This property is used to set or retrieve the type of border to be used when drawing tables.

Usage [*form.*]*vsPrinter*.**TableBorder** [= *setting%*]

Remarks Valid settings for this property are:
0 - None
1 - Bottom
2 - Top
3 - Top & Bottom
4 - Box
5 - Columns
6 - Columns, Top, & Bottom
7 - All
8 - Box Rows
9 - Box Columns

This property is analogous to the PageBorder property, except it applies to tables instead of pages.

Default Value 7 - All

Data Type Integer

TableSep Property

Description This property is used to specify the characters to be used as table row and column separators. It is used in conjunction with the Table property.

Usage [*form.*]*vsPrinter*.**TableSep** = *tablesep$*

Remarks TheTableSep property must hold a two-character string. The first character is used as the column separator and the second as the row separator. The characters must be different.

If the string is less than two characters long, or if the characters are equal, the default table separator string is used (|;).

Default Value |;

Data Type String

Text Property

Description This property is used to print text. The Printer control takes care of all the justification, word wrapping, and text flow.

Usage [*form.*]*vsPrinter*.**Text** = *text$*

Remarks This property is used to print pieces of paragraphs. This is useful if you want to change fonts or colors halfway through a left-aligned paragraph. For example, the code below prints VB files and uses bold red type for function names:

```
Sub PrintVBFile (FileName as String)
  Dim ln$, subname$, subargs$

  Open FileName For Input As #1
  vsPrinter.Action = 3                    start document
  While Not EOF (1)
    Line Input #1, ln

     print function header
    If Left(ln, 3) = Sub Then
      ln = Mid(ln, 4)
      subname = Left(ln, Instr (ln,  ))
      subargs = Mid(ln, Instr (ln,  ))
      vsPrinter.Text = Sub
      vsPrinter.FontBold = True
      vsPrinter.TextColor = RGB (255, 0, 0)
      vsPrinter.Text = subname
      vsPrinter.FontBold = False
      vsPrinter.TextColor = 0
      vsPrinter.Text = subargs

     print straight paragraph
    Else
      vsPrinter = ln
    End If
  Wend

  vsPrinter.Action = 6                    end document
End Sub
```

If you want to print an entire paragraph, use the Paragraph property instead.

Note that the text string may have embedded line breaks. For example, the code

```
vsPrinter.text = "1: Hello" & Chr$(13) & "World (2 lines)"
vsPrinter.text = "2: Hello "
vsPrinter.text = "World (1 line)"
```

would print

```
1: Hello
World (2 lines)
2: Hello World (1 line)
```

This property is write-only.

Data Type String

TextAlign Property

Description This property sets the alignment of paragraphs and tables.

Usage [*form.*]*vsPrinter*.**TextAlign** = *setting%*

Remarks Valid settings for this property are:
0 - Left
1 - Center
2 - Right

Tabs only work properly when text alignment is set to 0 (left).

Default Value 0 - Left

Data Type Integer

TextAngle Property

Description Specifies the angle, in tenths of degrees, between the base line of a character and the horizontal.

Usage [*form.*]*vsDraw*.**TextAngle** [= *angle%*]

Remarks The angle is measured in a counterclockwise direction.

Default Value 0

Data Type Integer

TextColor Property

Description This property sets or retrieves the color used to print regular text.

Usage [*form.*]*vsPrinter*.**TextColor** [= *color&*]

Remarks Use the HdrColor to define the color used to print header and footer text.

Default Value 0 (Black)

Data Type Long (Color)

TextHei, TextWid Properties

Description These properties are used to measure text, so you can do things such as draw boxes around strings and center text. They work in conjunction with the Measure property.

Usage *textwidth&* = [*form.*]*vsPrinter*.**TextHei**

Remarks To measure text, start by selecting the appropriate font through the Font* properties. Then, assign the text to be measured to the Measure property. You can then read the text width and height, in Twips, from the TextHei and TextWid properties.

The example below draws a string at the current cursor position with a box around it:

```
Sub BoxString (s As String)
  vsPrinter.Measure = s
  vsPrinter.X1 = vsPrinter.CurrentX
  vsPrinter.Y1 = vsPrinter.CurrentY
  vsPrinter.X2 = vsPrinter.CurrentX + vsPrinter.TextWid
  vsPrinter.Y2 = vsPrinter.CurrentY + vsPrinter.TextHei
  vsPrinter.Text = s                  draw text
  vsPrinter.Action = 2                draw box around it
End Sub
```

These properties are read-only.

Data Type Long (Twips)

Version Property

Description This property returns the version of the Printer control currently loaded in memory.

Usage *CheckVer%* = [*form.*]*vsPrint*.**Version**

Remarks You may want to check this value at the Form_Load event, to make sure the version that is executing is at least as current as the version used to develop your application.

The version number is a three digit integer where the first digit represents the major version number and the last two represent the minor version number. For example, version 3.5 would return 350.

This property is read-only.

Data Type Integer

X1, X2, Y1, Y2 Properties

Description These properties are used to control the graphics drawn using the Draw and Picture properties.

Usage [*form.*]*vsPrinter*.**X1** [= *value&*]

Remarks These properties define a rectangle. Units are Twips, measured from the top left corner of the page.

For an example of how these properties are used, see the description of the TextHei property.

Data Type Long (Twips)

Draw Reference

Description The VideoSoft Draw control lets you create detailed, scaleable, resolution-independent drawings. The drawings can be shown on your forms, printed, or copied to the Windows clipboard, where they become available for pasting into other Windows applications such as Word.

After the drawing is created, you can change the drawing extents to provide distortion and zoom.

Use the Draw control to create maps, charts, diagrams, or whatever graphics you need.

File Name VSVIEW.VBX

Object Type vsDraw

Remarks When you send drawing commands to the Draw control, they are stored in a list, but nothing is actually drawn on the screen or printer until you use the Action property to render the drawing. This is done this way to improve efficiency.

The Draw control is based on Windows metafile technology, which has limited support for text manipulation. This makes it less than ideal for text-intensive applications. If you need to handle a lot of text, use the Printer control instead.

Draw Summary

Properties (default: Action)

(About)	* Action	BackColor
* BackStyle	BorderStyle	* BrushColor
* BrushStyle	DragIcon	DragMode
* Draw	Enabled	FontBold
FontItalic	FontName	FontSize
FontStrike	FontUnder	Height
HelpContextID	Index	Left
MousePointer	Name	Parent
* PenColor	* PenStyle	* PenWidth
* Picture	* Polygon	* PolyLine
* ScaleHeight	* ScaleLeft	* ScaleTop
* ScaleWidth	Tag	* TextAlign
* TextAngle	* TextColor	* TextOut
Top	Visible	* Version
Width	* X1	* X2
* Y1	* Y2	

Events

Click	DblClick	DragDrop
DragOver	MouseDown	MouseMove
MouseUp		

Methods

Move	Refresh	SetFocus
ZOrder		

Action Property

Description This property allows you to specify actions to be taken by the Draw control.

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

Remarks Valid settings for this property are:
0 - None
1 - Clear
2 - Draw
3 - Print
4 - Choose Printer & Print
5 - Copy

The *Clear* action erases everything in the Draw control.

The *Draw* action renders the current drawing on the control. Note that you can create an entire drawing, but nothing will appear on the screen until you set the Action property to 2. This is done to improve rendering speed.

The *Print* action renders the drawing on the printer. The drawing is automatically scaled to fill as much of the page as possible, while preserving the aspect ratio of the drawing on the screen.

The *Choose Printer & Print* action is similar to the *Print* action, except it displays a dialog box that allows the user to choose the printer to be used.

The *Copy* action copies the current drawing to the Windows clipboard, where it becomes available to be pasted into other applications.

This property only acts at run-time, but the above list is also shown at design-time, on the properties window. This makes it easy for you to remember the code for each action.

This property is read-only.

Data Type Integer

BackStyle Property

Description This property determines whether text drawn on the Draw control is transparent or opaque.

Usage [*form.*]*vsDraw*.**BackStyle** [= *setting%*]

Remarks Valid settings for this property are:
0 - Transparent
1 - Opaque

Default Value 0 - Transparent

Data Type Integer

BrushColor Property

Description Sets or returns the color to be used when filling objects.

Usage [*form.*]*vsDraw*.**BrushColor** [= *colorref&*]

Remarks This property affects the drawing of rectangles and ellipses (see the Draw property) as well as polygons (see the Polygon property).

Default Value Black

Data Type Long (Color)

BrushStyle Property

Description Sets or returns the style to be used when filling objects.

Usage [*form.*]*vsDraw*.**BrushStyle** [= *colorref&*]

Remarks Valid settings for this property are:
0 - Solid
1 - Transparent
2 - Horizontal Line
3 - Vertical Line
4 - Upward Diagonal
5 - Downward Diagonal
6 - Cross
7 - Diagonal Cross

This property affects the drawing of rectangles and ellipses (see the Draw property) as well as polygons (see the Polygon property).

Default Value 0 - Solid

Data Type Long (Color)

Draw Property

Description This property allows you to draw lines, rectangles, and ellipses.

Usage [*form.*]*vsDraw*.**Draw** = *object%*

Remarks Valid settings for this property are:
0 - Nothing
1 - Line
2 - Rectangle
3 - Ellipse

All objects are drawn extending between the (x1, y2) and (x2, y2) points defined by the X1, Y1, X2, and Y2 properties.

Objects are drawn with the current pen and filled with the current brush. Pen and brush attributes are defined with the Pen* and Brush* properties.

This property only acts at run-time, but the above list is also shown at design-time, on the properties window. This makes it easy for you to remember the code for each object.

This property is write-only.

Data Type Integer

PenColor Property

Description This property sets or returns the color of the pen used to draw all graphics.

Usage [*form.*]*vsDraw*.**PenColor** [= *colorref&*]

Remarks This property does not affect text. To change text color, use the TextColor property.

Default Value 0 (Black)

Data Type Long (Color)

PenStyle Property

Description This property sets or returns the style of the pen used to draw all graphics.

Usage [*form.*]*vsDraw*.**PenStyle** [= *setting%*]

Remarks Valid settings for this property are:
0 - Solid
1 - Dash
2 - Dot
3 - Dash-Dot
4 - Dash-Dot-Dot
5 - Transparent
6 - Inside Solid

Note that non-solid styles only work when the PenWidth property is set to zero. This is a GDI limitation.

Default Value 0 -Solid

Data Type Integer

PenWidth Property

Description This property sets or returns the width of the pen used to draw all graphics.

Usage [*form.*]*vsDraw*.**PenWidth** [= *width%*]

Remarks Setting PenWidth to zero causes the thinnest possible pen to be used.

Default Value 0

Data Type Integer

Picture Property

Description This property returns a picture representing the contents of the Draw control.

Usage [*form.*]*vsPrint*.**Picture** = [*form.*]*vsDraw*.**Picture**

Remarks This property is mainly useful when you want to copy the contents of a Draw control directly into a Printer control.

This property is read-only.

Data Type Picture

PolyLine Property

Description This property allows you to plot a line composed of many points.

Usage [*form.*]*vsDraw*.**PolyLine** = *points$*

Remarks The *points$* string contains a sequence of X, Y coordinates, separated by spaces. Only the integer part of the coordinates is used. The coordinates are arbitrary, determined by the Scale* properties.

You could create complex lines by repeatedly setting the Draw property to *Line* and updating X1, Y1, X2, and Y2, but the PolyLine is usually a more convenient and efficient way of obtaining the same results.

The line is drawn with the current pen, defined by the Pen* properties.

This property is write-only

Data Type String

Polygon Property

Description This property allows you to plot an arbitrary filled polygon. It is similar to the PolyLine property except that it produces a closed figure.

Usage [*form.*]*vsDraw*.**Polygon** = *points$*

Remarks For details on the syntax of the *points$* parameter, see the description of the PolyLine property.

Data Type String

ScaleHeight, ScaleWidth Properties

Description These properties set or return the extents of the coordinate system used by the Draw control.

Usage [*form.*]*vsDraw*.**ScaleHeight** = *height%*

Remarks All drawing on the Draw control is done in an arbitrary coordinate system, determined by the Scale* properties.

For example, the following code draws a rectangle that fills the entire Draw control, regardless of its physical dimensions:

```
vsDraw.X1 = 0
vsDraw.Y1 = 0
vsDraw.X2 = 1000
vsDraw.Y2 = 1000
vsDraw.Draw = 2      Draw = Rectangle
vsDraw.Action = 2  Action = Draw
```

You can stretch drawings, zoom, or scroll by modifying the coordinate system and setting the Action property to 2 (Draw).

Default Value 1000

Data Type Integer

ScaleLeft, ScaleTop Properties

Description These properties set or return the logical origin of the coordinate system used by the Draw control.

Usage [*form.*]*vsDraw*.**ScaleLeft** = *left%*

Remarks All drawing on the Draw control is done in an arbitrary coordinate system, determined by the Scale* properties.

For example, the following code sets the (0,0) point of the logical coordinate system to the center of the Draw control, regardless of its physical dimensions:

```
vsDraw.ScaleLeft = -vsDraw.ScaleWidth / 2
vsDraw.ScaleTop = -vsDraw.ScaleHeight / 2
```

You can stretch drawings, zoom, or scroll by modifying the coordinate system and setting the Action property to 2 (Draw).

Default Value 1000

Data Type Integer

TextAlign Property

Description This property sets the alignment to be used when drawing text.

Usage [*form.*]*vsDraw*.**TextAlign** = *setting%*

Remarks Valid settings for this property are:
0 - Left
1 - Center
2 - Right

Default Value 0 - Left

Data Type Integer

TextAngle Property

Description Specifies the angle, in tenths of degrees, between the base line of a character and the horizontal.

Usage [*form.*]*vsDraw*.**TextAngle** [= *angle%*]

Remarks The angle is measured in a counterclockwise direction when the y direction is down and in a clockwise direction when the y direction is up.

Default Value 0

Data Type Integer

TextColor Property

Description This property sets or retrieves the color used to print text.

Usage [*form.*]*vsDraw*.**TextColor** [= *color&*]

Default Value 0 (Black)

Data Type Long (Color)

TextOut Property

Description Assigning a string to this property causes the Draw control to print the string at the current cursor position, determined by the X1 and Y1 properties.

Usage [*form.*]*vsDraw*.**TextOut** = *text$*

Remarks The cursor position is not updated after printing.

The string assigned to TextOut may have embedded carriage-returns (chr$(13)) characters, which cause line breaks.

This property is write-only.

Data Type String

Version Property

Description This property returns the version of the Draw control currently loaded in memory.

Usage *CheckVer%* = [*form.*]*vsDraw*.**Version**

Remarks You may want to check this value at the Form_Load event, to make sure the version that is executing is at least as current as the version used to develop your application.

The version number is a three digit integer where the first digit represents the major version number and the last two represent the minor version number. For example, version 3.5 would return 350.

This property is read-only.

Data Type Integer

X1, X2, Y1, Y2 Properties

Description These properties are used to control the graphics drawn using the Draw property and text drawn with the TextOut property.

Usage [*form.*]*vsDraw*.**X1** [= *value%*]

Remarks These properties define a rectangle. The coordinates are arbitrary, determined by the Scale* properties.

Data Type Integer

ViewPort Reference

Description The VideoSoft ViewPort is a scrollable container control. With ViewPort, you no longer have to write tedious code to synchronize scroll bars and picture boxes. You decide how big your window should be, then let the ViewPort scroll your controls for you.

File Name VSVIEW.VBX

Object Type vsViewPort

Remarks The ViewPort automatically scrolls all its windowed child controls, but it does not scroll graphical controls such as Labels and Image Boxes.

Use the ViewPort with the VideoSoft Print control to implement print preview in your programs.

ViewPort Summary

Properties (default: AutoScroll)

(About)	* AutoScroll	BackColor
BorderStyle	Height	HelpContextID
Hwnd	Index	* LargeChangeHorz
* LargeChangeVert	Left	Name
Parent	* SmallChangeHorz	* SmallChangeVert
Tag	Top	* Track
* Version	* VirtualHeight	* VirtualLeft
* VirtualTop	* VirtualWidth	Width

Events

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

AutoScroll Property

Description This property determines whether ViewPort should automatically scroll its contents when the user clicks the scroll bars.

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

Remarks This property should be set to True in most applications, so that scrolling is done automatically by the ViewPort control.

If you wish to process scrolling yourself, set AutoScroll to False and respond to the Scroll event.

Default Value True

Data Type Boolean

LargeChangeHorz, LargeChangeVert Properties

Description These properties determine the amount of change to the VirtualLeft and VirtualTop properties when the user clicks the area between the scroll box and scroll arrow. Units are Twips.

Usage [*form.*]*vsViewPort*.**LargeChangeHorz** [= *setting&*]

Default Value 300 Twips

Data Type Long

Scroll Event

Description Fired whenever the values of VirtualLeft and VirtualTop change.

Syntax **Sub** *vsViewPort_***Scroll ()**

Remarks If AutoScroll is set to True, the Scroll event is fired after the actual scrolling takes place. If AutoScroll is set to False, the Scroll event is fired anyway.

SmallChangeHorz, SmallChangeVert Properties

Description These properties determine the amount of change to the VirtualLeft and VirtualTop properties when the user clicks the scroll arrow. Units are Twips.

Usage [*form.*]*vsViewPort*.**SmallChangeHorz** [= *setting&*]

Default Value 30 Twips

Data Type Long

Track Property

Description This property determines whether ViewPort should perform scrolling while the user moves the scroll boxes.

Usage [*form.*]*vsViewPort*.**Track** [= {**True**|**False**}]

Remarks Setting Track to true provides more user feedback during scrolling, but it can slow down your application and cause some flicker. If in doubts, try it both ways and see which works best for you.

Default Value False

Data Type Boolean

VirtualHeight, VirtualWidth Properties

Description These properties determine the extent of the area that can be seen by scrolling, regardless of the actual size of the control. Units are Twips.

Usage [*form.*]*vsViewPort*.**VirtualHeight** [= *setting&*]

Remarks If the VirtualHeight is smaller than the actual height of the control, the ViewPort automatically hides the vertical scroll bar. The same applies to VirtualWidth and the horizontal scroll bar.

See also the description of the VirtualLeft and VirtualTop properties.

Default Value 0 Twips

Data Type Long

Version Property

Description This property returns the version of the ViewPort control currently loaded in memory.

Usage *CheckVer%* = [*form.*]*vsViewPort*.**Version**

Remarks You may want to check this value at the Form_Load event, to make sure the version that is executing is at least as current as the version used to develop your application.

The version number is a three digit integer where the first digit represents the major version number and the last two represent the minor version number. For example, version 3.5 would return 350.

This property is read-only.

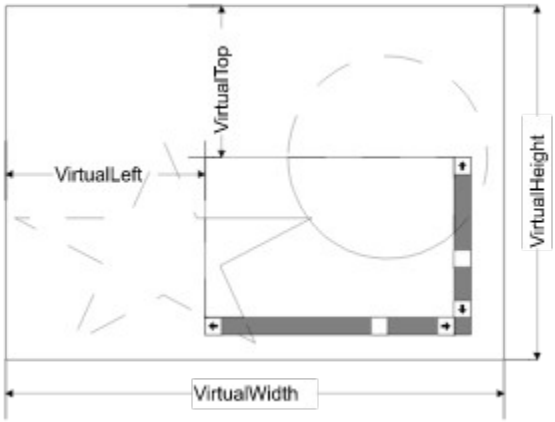
Data Type Integer

VirtualLeft, VirtualTop Properties

Description These properties determine what portion of the scrollable area is visible at any time. Units are Twips.

Usage [*form.*]*vsViewPort*.**VirtualLeft** [= *setting&*]

Remarks The diagram below illustrates the relationship between VirtualLeft/Top and VirtualWidth/Height:



If the AutoScroll property is set to True, then changes to these properties cause ViewPort to scroll its contents and to fire the Scroll event.

Default Value 0 Twips

Data Type Long

VideoSoft Products

Registration Form

VSVBX

A set of three custom controls for interface design and text parsing.

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	Allows you to group controls by subject, using the familiar notebook metaphor that has become a Windows standard.
	Awk	VsAwk	Parsing engine named and patterned after the popular Unix utility, plus a powerful expression evaluator.

VSVIEW

A set of four custom controls for creating, viewing, and printing text and graphics.

Icon	Name	Object	Description
	InForm	vsInForm	A control that you can drop into any container to customize its title bar, frame, resizing behavior, and frame buttons. InForm also allows you to monitor the clipboard, drag and drop files from File manager, and more.
	Printer	vsPrinter	A much improved printer object with word wrap, headers and footers, multi-column printing, graphics, and multi-page Print Preview capability.
	ViewPort	vsViewPort	A control that gives you a scrollable virtual area so you can fit more controls in your windows. Great for implementing Print Preview and programs that look like the Program Manager.
	Draw	vsDraw	A versatile drawing control that lets you create complex images, view them on the screen, copy them to the clipboard, or print them. Great for technical drawings, maps, and diagrams.

CODEBOOK

A handy, integrated suite of utilities to help you develop Visual Basic applications.

Registration Form

(You may print this form by selecting the File|Print command).

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

(To order by phone, call 1-800-5477295)

Please register my copy of the following VideoSoft products. I am enclosing a check or money order for the amount of:

VSVBX Single developer	US$ 45.00	
Additional developers	__ x 45.00	
VSVIEW Single developer	95.00	
Additional developers	__ x 95.00	
CODEBOOK Single developer	45.00	
Additional developers	__ x 45.00	
Shipping and Handling Domestic	6.00	
Shipping and Handling International	10.00	
CA Sales Tax (CA residents only)	8.5%	
TOTAL		US$

Note: Call us for details on site licenses and volume discounts.

Name:

Company:

Street:

City, State, ZIP:

Country:

Phone:

Where did you hear about the VideoSoft products?