VideoCapX control ActiveX control

By Fath Software

License agreement

This Limited Use Software License Agreement (the "Agreement") is a legal agreement between you, the end-user ("Licensee"), and author. By using this software or storing this program ("VideoCapX control") on a computer hard drive or other media), you are agreeing to be bound by the terms of this Agreement.

You may install trial version of this program to test and evaluate for 30 days; after that time you must either register this program or delete it from your computer hard drive.

The trial version of software may be distributed freely on online services, bulletin boards, or other electronic media as long as the files are distributed in their entirety.

You may not alter this software in any way, including changing or removing any messages or windows.

You may not decompile, reverse engineer, disassemble or otherwise reduce this software to a human perceivable form. You may not modify, rent or resell for profit this software. You may not publicize or distribute any registration code algorithms, information, or registration codes used by this software without permission of author.

Author grants a license to use the enclosed software to the original purchaser.
Customer written applications containing embedded VideoCapX control may be freely distributed, without royalty payments to author, provided that such distributed product is bound into these applications in such a way so as to prohibit separate use in design mode, and that such product is distributed only in conjunction with the customer's own software product. This control may be used as a constituent control only if the compound control thus created is distributed with and as an integral part of an application. This license may be transferred to a third party only if all existing copies of the software and its documentation are also transferred.

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

This software is provided "as is". Author makes no warranty, expressed or implied, with regard to the software. All implied warranties, including the warranties of merchantability and fitness for a particular use, are hereby excluded.
AUTHOR'S LIABILITY IS LIMITED TO THE PURCHASE PRICE. Under no circumstances shall author of this product be liable for any incidental or consequential damages, nor for any damages in excess of the original purchase price.

Plain English version:

We require that you purchase one copy of a control per developer on a project. If this is met, you may distribute the control with your application royalty free. You may never distribute the LIC file. You may not change the product in any way that removes or changes the requirement of a license file.
We encourage the use of our controls as constituent controls when the compound controls you create are an integral part of your application. But we don't allow distribution of our controls as constituents of other controls when the compound control is not part of an application. The reason we need to have this restriction is that without it someone might decide to use our control as a constituent, add some trivial (or even non-trivial) enhancements and then sell the compound control. Obviously there would be little difference between that and just plain reselling our control.

If you have purchased the source code, you may not re-distribute the source code either (nor may you copy it into your own project). Author retains the copyright to the source code.
Your license is transferable. The original purchaser of the product must make the transfer request. Contact us for further information.
The sample versions of our products are intended for evaluation purposes only. You may not use the sample version to develop completed applications.

Technical support

Internet Mail

You can send E-mail to technical support via the Internet. Messages should be addressed to support@fathsoft.com .

World Wide Web

The Fath Software web site is located at http://www.fathsoft.com . You can access our web site for up-to-date information about the product, support forum, knowledgebase and news.

VideoCapX control reference

vcxCKFlags	Chroma key flags
vcxStretchModeEnum	Stretch mode values
vcxUseDeinterlaceEnum	UseDeinterlace values
vcxUseVideoFilterEnum	UseVideoFilter values
vcxVideoRendererEnum	VideoRenderer values
AudioCodecIndex	Set index of audio codec to use for audio compression
AudioDeviceIndex	Set index of audio device to use for capturing audio
AudioInputIndex	Specify input port for audio on multi-port audio input cards
BackColor	property BackColor
CapFilename	Filename for captured media file. Extension can be AVI or WMV.
CapTimeLimit	Time limit for capturing, in seconds
CapTimeLimitEnabled	Indicate is CapTimeLimit property valid
CaptureAudio	Indicate will audio be captured
CaptureBufferLength	Sets video buffer length in seconds. Use SaveBuffer method to save this buffer into a video file.
CaptureRate	Get/Set video capture rate (number of frames per second)
ColorFormat	Specify color format of the source video stream
Connected	Get/set connection to video device
DebugMode	Internal. Do not use.
DeviceType	Returns connected video device type. 0,1,2,3 for unknown, TV tuner, DV camera, DV VCR .
EnableNewFrameEvent	If set to TRUE, every new video frame will generate NewFrame event
FTPPassiveMode	If set to TRUE, FTP transfer methods will use passive mode.
HasOverlay	(Read-only) Returns TRUE if selected video device supportes video overlay feature
hWnd	Returns Windows window handle of VideoCapX control
IsCapturing	Returns TRUE if video-capture is in progress
LocalAddress	If mulitple network adapters are installed, this property specifies which one to use in network comunication.
MasterStream	Specify master stream in AVI file (audio or video or none)
MouseIcon	Set custom mouse icon
MousePointer	Set mouse pointer shape
Overlay	Enable/disable video operlay preview
Overscan	Specify how many pixels to discard at video borders.
Preview	Enable/disable video preview

PreviewAudio	Set this property to TRUE if you want audio in preview mode
PreviewFullScreen	When set to TRUE, preview video will cover the whole screen
ProfileData	Set custom WM profile XML data
ProfileIndex	Specify system profile to use when creating WMV files
RelayServer	property RelayServer
RelayUsername	property RelayUsername
ResizeBroadcast	resize broadcast video to specified ratio
ResizeCapture	Resize captured video using specified ratio, 2 for double, 0.5 for half size.
ScaleVideo	property ScaleVideo
ServerMode	If set to TRUE, the control will listed for TCP connections on ServerPort mode and send video frames
ServerPassword	Sets password for server access
ServerPort	Number of TCP port for ServerMode
ServerQuality	Specify quality of video images transfered by ReceiveFrame method. Range 10-100. Defult 30.
StretchMode	Control how video will be resized
SyncUsingStreamOffset	Is stream offset used to synchronize audio/video streams in captured file
UseDeinterlace	Deinterlace video
UseOverlay	If set to TRUE, VideoCapX will output video preview to overlay surface of graphics adapter.
UserFilter2CLSID	Second user filter
UserFilter3CLSID	Third user filter
UserFilterCLSID	Specify user video filter by CLSID
UserFilterIUnknown	Set IUnknown pointer of custom video filter.
UseVideoFilter	Determine if VideoCapX video filter will be used. This filter handles frame grabbing, video cropping and text/bitmap overlay. Without it, video stream can be much faster.
Version	Returns VideoCapX.OCX version number
VideoCodecIndex	Set index of video codec to use for on-the-fly compression of video
VideoCodecQuality	Sets quality parameter for video codec
VideoDeviceIndex	Set index of video device to use for capture
VideoFlip	Sets video flipping. Flips video image horizontally and/or vertically.
VideoHeight	Returns current video height in pixels. This property is read-only.
VideoInputIndex	Sets channel to use on multi-port capture cards
VideoRenderer	Select video renderer to use for video preview.
VideoSourceURL	URL of network camera acting as video source

VideoWidth	Returns current video width in pixels. This property is read-only.
WMAttributes	Windows Media attributes to set when capturing into WMV files or broadcasting
WMTVersion	Sets WindowsMedia system profiles version to use. See ProfileIndex property. Deafult is 7. Possible values are 4,7 and 8.
AboutBox	Shows About Box of VideoCapX
Acquire	Acquires an image from TWAIN source (like scanner).
AllocCapFile	Pre-allocates space on disk for capture file
AutoTune	The AutoTune method scans for a precise signal on the channel's frequency.
CameraControlGet	Returns value of camera-control properties
CameraControlGetRange	Retrieve camera control range values.
CameraControlSet	Sets camera-control properties
CompareImages	Returns difference between two images.
ConnectionState	returns current DisplayRemote connection state
CopyCaptureFile	Copies AVI file from pre-allocated storage into new file
CopyFrame	Copy current vide frame into clipboard
DetectMotion	Detect changes in video frames
DetectObjects	Detects moving objects on the image and returns their coordinates
DisplayRemote	Starts a video-cenference call
ExportToDV	Export DV-compatible AVI file to DV VTR device
GetActualFrameRate	Returns current actual frame rate. Some devices may provide lower frame rates than requested because of bandwidth availability. This is only available during video streaming.
GetAudioCodecCount	Returns installed audio codec count
GetAudioCodecName	Returns audio codec name
GetAudioDeviceCount	Returns number of audio devices in the system
GetAudioDeviceName	Returns audio device name
GetAudioFormat	Returns audio format parameters
GetAudioInputCount	Returns number of input ports on audio source
GetAudioInputName	Returns audio input port name
GetAudioLevel	Returns audio level in preview mode
GetAudioLevel2	Get audio levels for left and right channel
GetCapFileSize	Returns file size (in bytes) of capture file
GetCapInfo	method GetCapInfo
GetCapStatus	Retrieves video capture parameters
GetControlRef	Get control reference pointer
GetDateCode	Returns time on DV video tape (the time when the video has

	been taken)
GetDeviceDesc	Returns device description string
GetDeviceID	Returns unique device ID string
GetFilterSettings	Returns current filter settings
GetFrameAsHBITMAP	Return Windows HBITMAP value of current video frame.
GetProfileCount	Returns number of WindowsMedia system profiles
GetProfileDesc	Returns WindowsMedia profile description
GetProfileName	Returns WindowsMedia profile name
GetRGB	Returns current video frame image as array of RGB values.
GetTimecode	Return timecode value on digital VCR video type
GetTunerSignal	Returns 1 if TV signal is present on current channel
GetVideoCaps	Returns an array of supported video formats
GetVideoCodecCount	Returns installed video codec count
GetVideoCodecName	Returns video codec name.
GetVideoDeviceCount	Returns number of video-capture devices in the system
GetVideoDeviceDesc	Returns video device description
GetVideoDeviceName	Returns video device name
GetVideoFormat	Returns video size.
GetVideoInputCount	Returns number of video inputs on currently selected video device (card).
GetVideoInputName	Returns name of specified video channel on multiple-input capture cards.
GetVideoProcAmp	Get video properties
GetVideoProcAmpValueRange	Retrieve value range for video property.
GetVRIUnknown	Returns IUnknown interface of current video renderer filter. See 'VideoRenderer' property.
GrabFrame	Returns current video frame as VB Picture object
HTTPUpload	Use HTTP upload protocol to send information and files to web server
LoadProfileFromURL	Loads WM profile data from .prx file. URL argument must start with 'http://' or 'file://' . Return value is 1 on success or 0 on failure.
PauseCapture	Pause capture
Ping	method Ping
PlayerClose	Closes media playback.
PlayerGetFrame	Returns current video frame number
PlayerGetFrameCount	Returns number of fideo frames in the movie
PlayerGetLenMS	Returns media file length in milliseconds.
PlayerGetPos	Returns current playing position.

PlayerGetVideoSize	Returns video size for media player.
PlayerOpen	Opens media file for playback.
PlayerOpenDVD	Open DVD volume
PlayerSetFrame	Sets current video frame
PlayerSetFullScreen	Opens full-screen playback.
PlayerSetMute	Mutes sound of media player.
PlayerSetPos	Sets current position for playback.
PlayerSetRate	Sets the playback rate
PlayerSetSize	Sets media player window size.
PlayerStart	Start playing media file.
PlayerStepFrames	Steps nFrames in player mode
PlayerStepOneFrame	Display next frame
PlayerStop	Stop playing media file.
PlayRemoteAudio	Connects to remote VideoCapX server and receives only audio stream.
ReceiveAudio	Receive audio data packet from server
ReceiveFrame	Return video frame from remote server as Picture object.
Recompress	Copies AVI/WMV into new file using specified video compression.
RecompressEx	Use this method to merge video and audio files and/or crop video files.
ResumeCapture	Resume capture paused with PauseCapture method
SaveBuffer	Saves current video buffer into a file with a name specified by CapFilename property and using video/audio codecs specified with VideoCodecIndex/AudioCodecIndex.
SaveFrame	Saves video image into file
SaveFrameJPG	Saves current video frame into JPG file
SavePictureJPG	Saves Picture object into JPG file.
SelectSource	Let the user select TWAIN source.
SendScriptCommand	Send script type/command pair to broadcast client. This method works only if WM broadcast started with StartBroadcast method is running.
SetAudioBalance	Sets the balance of the audio signal
SetAudioDelay	Sets audio delay (positive or negative) in captured AVI file. DelayMS argument is in milliseconds.
SetAudioFormat	Sets audio format for capture
SetAudioInputLevel	Sets the recording level for audio input selected with AudioInputIndex property. Level value is in range 0 to 100.
SetAudioVolume	Sets audio renderer volume. Volume argument range is 0 - 100. if Volume is -1, this method returns current audio volume.

SetBitmapOverlay	Sets bitmap to show on-video
SetChromaKey	Sets chroma-key effect. BackImage parameter is a filename of static image background which will be visible through video. Color parameters are RGB values.
SetCrop	Crops live-video stream to the rectangle of dimensions (W,H) and with top-left coordinate of (X,Y).
SetFadeLevel	Sets fade level (the range is 0-100, 0 is neutral) for video
SetFilterSettings	Set filter settings
SetHighPriority	Sets priority class for the current process. High argument is TRUE for high priority or FALSE for normal priority.
SetMasterAudioVolume	Set master volume for specified mixer line. Volume argument range is 0 - 100. if Volume is -1, this method returns current audio volume.
SetMotionMask	Sets rectangle(s) to ignore on the image while detecting motion.
SetOption	method SetOption
SetSlaveControl	Set video source control
SetTextOverlay	Sets on-video text caption.
SetTunerChannel	Set TV tuner channel
SetTunerCountryCode	Set/Get TV tuner country code
SetTunerInputType	Set / retrieve TV tuner input type; cable or antenna.
SetTunerMode	Sets a multifunction tuner to the specified mode.
SetTVFormat	Set TV format (PAL, NTSC, etc.)
SetVideoFormat	Set video image dimensions
SetVideoFormatEx	Sets video format by capability index returned by GetVideoCaps method
SetVideoProcAmp	Set video properties
SetZoom	Set zoom rectangle on video. Use all zeros as parameters to this method to reset zoom.
ShowAudioCodecDlg	Shows audio codec dialog.
ShowAudioFormatDlg	Shows audio format dialog.
ShowAudioSourceDlg	Shows audio source dialog.
ShowTunerDlg	Display TV tuner properties dialog
ShowUserFilterDlg	Display user filter property page
ShowVideoCodecDlg	Shows video codec dialog.
ShowVideoCrossbarDlg	Display video crossbar dialog
ShowVideoFormatDlg	Shows video format dialog.
ShowVideoSourceDlg	Shows video source dialog.
SingleFrameAdd	Adds current video frame into AVI file opened by SingleFrameOpen method
SingleFrameAddPicture	Adds a Windows bitmap to the AVI file created with

	SingleFrameAdd method.
SingleFrameClose	Closes single-frame AVI capture
SingleFrameOpen	Creates AVI file for single-frame capturing.
StartBroadcast	Starts WindowsMedia network broadcast at specified port. Use Windows MediaPlayer's OpenURL command to see video on network.
StartBroadcastPush	Start sending broadcast to Windows Media server publishing point
StartCapture	Starts video capture
StopBroadcast	Stops WM broadcast
StopCapture	Stops video capture
StoreAutoTune	The StoreAutoTune method saves the fine-tuning information for all channels.
UploadFile	Upload a file to FTP server
UploadFrame	Sends current video frame to FTP server
VCRSetMode	Control Digital Video VCR. Use this method to change VCR modes of DV camcorder.
CaptureEnd	Triggered when capture is ended
CaptureReady	Raised after StartCapture is called but before any video is actually captured into file. It gives application the opportunity to display 'press to start capture...' message.
CaptureStart	Triggered when capture is started
DeviceLost	Raised when device lost is detected. Such as camera removal or cable plug-out.
FullscreenLost	Raised when full-screen mode ends due to user action
NewFrame	Raised when new video frame is available
RecompressCompleted	Recompress method runs in background. This event is fired when recompression is finished.
RecompressProgress	Reports progress of Recompress method processing.

Constants

DVD_AUDIO_APPMODE

DVD_AudioMode_None	0
DVD_AudioMode_Karaoke	1
DVD_AudioMode_Surround	2
DVD_AudioMode_Other	3

DVD_AUDIO_FORMAT

DVD_AudioFormat_AC3	0
DVD_AudioFormat_MPEG1	1

DVD_AudioFormat_MPEG1_DRC 2
DVD_AudioFormat_MPEG2 3
DVD_AudioFormat_MPEG2_DRC 4
DVD_AudioFormat_LPCM 5
DVD_AudioFormat_DTS 6
DVD_AudioFormat_SDDS 7
DVD_AudioFormat_Other 8

DVD_AUDIO_LANG_EXT

DVD_AUD_EXT_NotSpecified 0
DVD_AUD_EXT_Captions 1
DVD_AUD_EXT_VisuallyImpaired 2
DVD_AUD_EXT_DirectorComments1 3
DVD_AUD_EXT_DirectorComments2 4

DVD_CMD_FLAGS

DVD_CMD_FLAG_None 0
DVD_CMD_FLAG_Flush 1
DVD_CMD_FLAG_SendEvents 2
DVD_CMD_FLAG_Block 4
DVD_CMD_FLAG_StartWhenRendered 8
DVD_CMD_FLAG_EndAfterRendered 16

DVD_DISC_SIDE

DVD_SIDE_A 1
DVD_SIDE_B 2

DVD_DOMAIN

DVD_DOMAIN_FirstPlay 1
DVD_DOMAIN_VideoManagerMenu 2
DVD_DOMAIN_VideoTitleSetMenu 3
DVD_DOMAIN_Title 4
DVD_DOMAIN_Stop 5

DVD_FRAMERATE

DVD_FPS_25 1
DVD_FPS_30NonDrop 3

DVD_KARAOKE_ASSIGNMENT

DVD_Assignment_reserved0 0
DVD_Assignment_reserved1 1

DVD_Assignment_LR	2
DVD_Assignment_LRM	3
DVD_Assignment_LR1	4
DVD_Assignment_LRM1	5
DVD_Assignment_LR12	6
DVD_Assignment_LRM12	7

DVD_KARAOKE_CONTENTS

DVD_Karaoke_GuideVocal1	1
DVD_Karaoke_GuideVocal2	2
DVD_Karaoke_GuideMelody1	4
DVD_Karaoke_GuideMelody2	8
DVD_Karaoke_GuideMelodyA	16
DVD_Karaoke_GuideMelodyB	32
DVD_Karaoke_SoundEffectA	64
DVD_Karaoke_SoundEffectB	128

DVD_KARAOKE_DOWNMIX

DVD_Mix_0to0	1
DVD_Mix_1to0	2
DVD_Mix_2to0	4
DVD_Mix_3to0	8
DVD_Mix_4to0	16
DVD_Mix_Lto0	32
DVD_Mix_Rto0	64
DVD_Mix_0to1	256
DVD_Mix_1to1	512
DVD_Mix_2to1	1024
DVD_Mix_3to1	2048
DVD_Mix_4to1	4096
DVD_Mix_Lto1	8192
DVD_Mix_Rto1	16384

DVD_MENU_ID

DVD_MENU_Title	2
DVD_MENU_Root	3
DVD_MENU_Subpicture	4
DVD_MENU_Audio	5
DVD_MENU_Angle	6
DVD_MENU_Chapter	7

DVD_OPTION_FLAG

DVD_ResetOnStop	1
DVD_NotifyParentalLevelChange	2
DVD_HMSF_TimeCodeEvents	3
DVD_AudioDuringFFwdRew	4

DVD_PARENTAL_LEVEL

DVD_PARENTAL_LEVEL_8	32768
DVD_PARENTAL_LEVEL_7	16384
DVD_PARENTAL_LEVEL_6	8192
DVD_PARENTAL_LEVEL_5	4096
DVD_PARENTAL_LEVEL_4	2048
DVD_PARENTAL_LEVEL_3	1024
DVD_PARENTAL_LEVEL_2	512
DVD_PARENTAL_LEVEL_1	256

DVD_PREFERRED_DISPLAY_MODE

DISPLAY_CONTENT_DEFAULT	0
DISPLAY_16x9	1
DISPLAY_4x3_PANSCAN_PREFERRED	2
DISPLAY_4x3_LETTERBOX_PREFERRED	3

DVD_RELATIVE_BUTTON

DVD_Relative_Upper	1
DVD_Relative_Lower	2
DVD_Relative_Left	3
DVD_Relative_Right	4

DVD_SUBPICTURE_CODING

DVD_SPCoding_RunLength	0
DVD_SPCoding_Extended	1
DVD_SPCoding_Other	2

DVD_SUBPICTURE_LANG_EXT

DVD_SP_EXT_NotSpecified	0
DVD_SP_EXT_Caption_Normal	1
DVD_SP_EXT_Caption_Big	2
DVD_SP_EXT_Caption_Children	3
DVD_SP_EXT_CC_Normal	5
DVD_SP_EXT_CC_Big	6
DVD_SP_EXT_CC_Children	7

DVD_SP_EXT_Forced	9
DVD_SP_EXT_DirectorComments_Normal	13
DVD_SP_EXT_DirectorComments_Big	14
DVD_SP_EXT_DirectorComments_Children	15

DVD_SUBPICTURE_TYPE

DVD_SPType_NotSpecified	0
DVD_SPType_Language	1
DVD_SPType_Other	2

DVD_TextCharSet

DVD_CharSet_Unicode	0
DVD_CharSet_ISO646	1
DVD_CharSet_JIS_Roman_Kanji	2
DVD_CharSet_ISO8859_1	3
DVD_CharSet_ShiftJIS_Kanji_Roman_Katakana	4

DVD_TextStringType

DVD_Struct_Volume	1
DVD_Struct_Title	2
DVD_Struct_ParentalID	3
DVD_Struct_PartOfTitle	4
DVD_Struct_Cell	5
DVD_Stream_Audio	16
DVD_Stream_Subpicture	17
DVD_Stream_Angle	18
DVD_Channel_Audio	32
DVD_General_Name	48
DVD_General_Comments	49
DVD_Title_Series	56
DVD_Title_Movie	57
DVD_Title_Video	58
DVD_Title_Album	59
DVD_Title_Song	60
DVD_Title_Other	63
DVD_Title_Sub_Series	64
DVD_Title_Sub_Movie	65
DVD_Title_Sub_Video	66
DVD_Title_Sub_Album	67
DVD_Title_Sub_Song	68

DVD_Title_Sub_Other	71
DVD_Title_Orig_Series	72
DVD_Title_Orig_Movie	73
DVD_Title_Orig_Video	74
DVD_Title_Orig_Album	75
DVD_Title_Orig_Song	76
DVD_Title_Orig_Other	79
DVD_Other_Scene	80
DVD_Other_Cut	81
DVD_Other_Take	82

DVD_TIMECODE_FLAGS

DVD_TC_FLAG_25fps	1
DVD_TC_FLAG_30fps	2
DVD_TC_FLAG_DropFrame	4
DVD_TC_FLAG_Interpolated	8

DVD_TITLE_APPMODE

DVD_AppMode_Not_Specified	0
DVD_AppMode_Karaoke	1
DVD_AppMode_Other	3

DVD_VIDEO_COMPRESSION

DVD_VideoCompression_Other	0
DVD_VideoCompression_MPEG1	1
DVD_VideoCompression_MPEG2	2

VALID_UOP_FLAG

UOP_FLAG_Play_Title_Or_AtTime	1
UOP_FLAG_Play_Chapter	2
UOP_FLAG_Play_Title	4
UOP_FLAG_Stop	8
UOP_FLAG_ReturnFromSubMenu	16
UOP_FLAG_Play_Chapter_Or_AtTime	32
UOP_FLAG_PlayPrev_Or_Replay_Chapter	64
UOP_FLAG_PlayNext_Chapter	128
UOP_FLAG_Play_Forwards	256
UOP_FLAG_Play_Backwards	512
UOP_FLAG_ShowMenu_Title	1024
UOP_FLAG_ShowMenu_Root	2048

UOP_FLAG_ShowMenu_SubPic	4096
UOP_FLAG_ShowMenu_Audio	8192
UOP_FLAG_ShowMenu_Angle	16384
UOP_FLAG_ShowMenu_Chapter	32768
UOP_FLAG_Resume	65536
UOP_FLAG_Select_Or_Activate_Button	131072
UOP_FLAG_Still_Off	262144
UOP_FLAG_Pause_On	524288
UOP_FLAG_Select_Audio_Stream	1048576
UOP_FLAG_Select_SubPic_Stream	2097152
UOP_FLAG_Select_Angle	4194304
UOP_FLAG_Select_Karaoke_Audio_Presentation_Mode	8388608
UOP_FLAG_Select_Video_Mode_Preference	16777216

vcxCKFlags

vcxCkFront 0
vcxCkBack 1

vcxStretchModeEnum

None 0
Stretch 1
KeepAspectRatio 2

vcxUseDeinterlaceEnum

vcxNone 0
vcxSimple 1
vcxBob 2
vcx50fps 4

vcxUseVideoFilterEnum

vcxNo 0
vcxBoth 1
vcxPreviewOnly 2
vcxGrabOnly 3
vcxCaptureOnly 4

vcxVideoRendererEnum

vcxSystemDefaultRenderer 0
vcxVMR9 1
vcxVMR7 2
vcxGDI 3

vcxDeckLinkVideoRenderer 4
vcxOverlaySurfaceRenderer 10
vcxEVR 11

Properties

AudioCodecIndex

Property AudioCodecIndex As Long

Set index of audio codec to use for audio compression

AudioDeviceIndex

Property AudioDeviceIndex As Long

Set index of audio device to use for capturing audio

If this property is set to value of 100 and CaptureAudio is TRUE, VideoCapX uses video device for audio capturing also.

AudioInputIndex

Property AudioInputIndex As Long

Specify input port for audio on multi-port audio input cards

BackColor

Property BackColor As Long

property BackColor

CapFilename

Property CapFilename As String

Filename for captured media file. Extension can be AVI or WMV.

The default capture filename is "c:\capture.avi".
To split captured movie into several files, just change this property to a new filename while capture is running.
If capture filename ends with ".WMV", WindowsMedia format is used.

CapTimeLimit

Property CapTimeLimit As Long

Time limit for capturing, in seconds

The default value is 60.
This property will work only if CapTimeLimitEnabled is set to TRUE.

CapTimeLimitEnabled

Property CapTimeLimitEnabled As Boolean

Indicate is CapTimeLimit property valid

The default value for this property is FALSE.
If this value is TRUE video capture will stop after CapTimeLimit seconds.

CaptureAudio

Property CaptureAudio As Boolean

Indicate will audio be captured

The default value is TRUE.
If this value is FALSE, captured video will have no sound.

CaptureBufferLength

Property CaptureBufferLength As Double

Sets video buffer length in seconds. Use SaveBuffer method to save this buffer into a video file.

CaptureRate

Property CaptureRate As Double

Get/Set video capture rate (number of frames per second)

The default value is 15. CaptureRate property, to have an effect, must be set before connecting

capture driver.

ColorFormat

Property ColorFormat As Long

Specify color format of the source video stream

Values are:
0 = RGB24 (default),
1 = CLPL,
2 = YUYV,
3 = IYUV,
4 = YVU9,
5 = Y411,
6 = Y41P,
7 = YUY2,
8 = YVYU,
9 = UYVY,
10 = Y211,
11 = YV12,
12 = CLJR,
13 = IF09,
14 = CPLA,
15 = MJPG,
16 = TVMJ,
17 = WAKE,
18 = CFCC,
19 = IJPG,
20 = Plum,
21 = DVCS,
22 = DVSD,
23 = MDVF,
24 = RGB1,
25 = RGB4,
26 = RGB8,
27 = RGB565,
28 = RGB555,
29 = RGB32

VideoCapX.Connected = True
VideoCapX.ColorFormat = 3
VideoCapX.StartCapture

Connected

Property Connected As Boolean

Get/set connection to video device

Connects VideoCapX control onto video capture device specified with VideoDeviceToUse property.

VideoCapX.Connected = True

DebugMode

Property DebugMode As Long

Internal. Do not use.

DeviceType

Property DeviceType As Long

Returns connected video device type. 0,1,2,3 for unknown, TV tuner, DV camera, DV VCR .

Values are:
0,1,2,3 for unknown, TV tuner, DV camera, DV VCR .

EnableNewFrameEvent

Property EnableNewFrameEvent As Boolean

If set to TRUE, every new video frame will generate NewFrame event

FTPPassiveMode

Property FTPPassiveMode As Boolean

If set to TRUE, FTP transfer methods will use passive mode.

Passive FTP mode is firewall / proxy friendly.

HasOverlay

Property HasOverlay As Boolean

(Read-only) Returns TRUE if selected video device supportes video overlay feature

hWnd

Property hWnd As Long

Returns Windows window handle of VideoCapX control

IsCapturing

Property IsCapturing As Boolean

Returns TRUE if video-capture is in progress

LocalAddress

Property LocalAddress As String

If mulitple network adapters are installed, this property specifies which one to use in network comunication.

MasterStream

Property MasterStream As Long

Specify master stream in AVI file (audio or video or none)

Values are:
-1 = none
0 = video
1 = audio (default)

MouseIcon

Property MouseIcon As stdole.Picture

Set custom mouse icon

MousePointer

Property MousePointer As Long

Set mouse pointer shape

Mouse pointer values are:
0 (Default) Shape determined by the object.
1 Arrow.
2 Cross (crosshair pointer).
3 I beam.
4 Icon (small square within a square).
5 Size (four-pointed arrow pointing north, south, east, and west).
6 Size NE SW (double arrow pointing northeast and southwest).
7 Size N S (double arrow pointing north and south).
8 Size NW SE (double arrow pointing northwest and southeast).
9 Size W E (double arrow pointing west and east).
10 Up Arrow.
11 Hourglass (wait).
12 No Drop.
13 Arrow and hourglass.
14 Arrow and question mark.
15 Size all.

Overlay

Property Overlay As Boolean

Enable/disable video operlay preview

If HasOverlay property is TRUE, Overlay can be used instead of Preview property to show live video on-screen. Overlay video doesn't travel through main memory and it consumes almost no CPU resources. However, frame grabbing doesn't work. To be able to capture still frames and put text/bitmap over the video, you must use Preview property.
Connected proprety must be set to TRUE before you can use this property.

VideoCapX.Connected = True
VideoCapX.Overlay = True

Overscan

Property Overscan As Long

Specify how many pixels to discard at video borders.

Preview

Property Preview As Boolean

Enable/disable video preview

If set to True, starts video preview in VideoCapX window. Must be connected to use this property.

VideoCapX.Preview = True

PreviewAudio

Property PreviewAudio As Boolean

Set this property to TRUE if you want audio in preview mode

This property should be set before Preview=True.

PreviewFullScreen

Property PreviewFullScreen As Boolean

When set to TRUE, preview video will cover the whole screen

PreviewFullScreen property should be set to TRUE after form (window) with VideoCapX is shown on screen.

ProfileData

Property ProfileData As String

Set custom WM profile XML data

Custom profiles for WMV capture (for example: high quality video stream of 10Mbps) can be created with Windows Media Profile Editor tool included with MS Media Encoder9 . Custom profiles are saved into .prx files. PRX files are in plain-text XML format. You can copy XML data to ProfileData property to use this custom profile while capturing to WMV file.

ProfileIndex

Property ProfileIndex As Long

Specify system profile to use when creating WMV files

A profile is a collection of data that describes the configuration of an WMV file. The stream information in a profile contains the bit rate, buffer window, and media properties for the stream. The stream information for audio and video describes exactly how the media is configured in the file, including which codec (if any) will be used to compress the data.

You can create custom profiles with ProfileEdit tool of WMFormat SDK. To use custom profile,

set ProfileData property.

Use GetProfileCount, GetProfileName and GetProfileDesc methods to enumerate system profiles.

VideoCapX.Connected=True
VideoCapX.Preview=True
VideoCapX.CapFilename="capture.wmv"
VideoCapX.ProfileIndex=26
VideoCapX.StartCapture

RelayServer

Property RelayServer As String

property RelayServer

In order to improve broadcsating performance and client expirience, a relay server can be used. VideoCapX video server sends just one video stream to relay server and relay server with high bandwidth internet connected then sends that stream to all connected cilents.

For more info on how to obtain relay server, contact support@fathsoft.com .

' on server side
vcxsrv.Connected=True
vcxsrv.RelayServer="192.168.1.123"
vcxsrv.RelayUsername = "myServer"
vcxsrv.ServerPassword="pass"
vcxsrv.ServerMode=True
vcxsrv.Preview=True

'on client side
cln.RelayServer="192.168.1.123"
cln.RelayUsername = "client"
vcxsrv.ServerPassword="pass"
cln.DisplayRemote "myServer",1

RelayUsername

Property RelayUsername As String

property RelayUsername

ResizeBroadcast

Property ResizeBroadcast As Single

resize broadcast video to specified ratio

ResizeCapture

Property ResizeCapture As Single

Resize captured video using specified ratio, 2 for double, 0.5 for half size.

ScaleVideo

Property ScaleVideo As Double

property ScaleVideo

Set this property to 1.0 for original video size, 2.0 for double size and 0.5 for half-sized video on screen.

ServerMode

Property ServerMode As Boolean

If set to TRUE, the control will listed for TCP connections on ServerPort mode and send video frames

If set to TRUE, the control will listed for TCP connections on ServerPort mode and send video frames

ServerPassword

Property ServerPassword As String

Sets password for server access

ServerPort

Property ServerPort As Long

Number of TCP port for ServerMode

ServerQuality

Property ServerQuality As Long

Specify quality of video images transfered by ReceiveFrame method. Range 10-100. Defult 30.

StretchMode

Property StretchMode As vcxStretchModeEnum

Control how video will be resized

none = video is not rsized
stretch = video is resized to fit control rectangle without preserving aspect ratio
zoom = video is resized to fit control and preserver aspect ratio

SyncUsingStreamOffset

Property SyncUsingStreamOffset As Boolean

Is stream offset used to synchronize audio/video streams in captured file

UseDeinterlace

Property UseDeinterlace As vcxUseDeinterlaceEnum

Deinterlace video

Possible values are:
vcxNone=0 - no deinterlacing
vcxSimple=1 - simple (fast) deinterlacing
vcxBob=2 - Better deinterlacing

UseVideoFilter must be enabled for deinterlacing to work.

UseOverlay

Property UseOverlay As Boolean

If set to TRUE, VideoCapX will output video preview to overlay surface of graphics adapter.

UserFilter2CLSID

Property UserFilter2CLSID As String

Second user filter

UserFilter3CLSID

Property UserFilter3CLSID As String

Third user filter

UserFilterCLSID

Property UserFilterCLSID As String

Specify user video filter by CLSID

Set this property to CLSID string of custom video filter you want to use. It should be set before setting Connected to TRUE.
VideoCapX will create instance of this CLSID and release it after Conneced is set to FALSE.

VideoCapX.UserFilterCLSID = "{c200e360-38c5-11ce-ae62-08002b2b79ef}"
VideoCapX.Connected = TRUE
VideoCapX.Preview = TRUE

UserFilterIUnknown

Property UserFilterIUnknown As Unknown

Set IUnknown pointer of custom video filter.

You must create the filter, set it's properties and QueryInterface for IUnknown. Then, set this property to your filter IUnknown pointer before setting Connected property of VideoCapX to TRUE.
VideoCapX doesn't call AddRef or Release on this interface. It will QueryInterface for IBaseFitler and call AddRef/Release on it.

UseVideoFilter

Property UseVideoFilter As vcxUseVideoFilterEnum

Determine if VideoCapX video filter will be used. This filter handles frame grabbing, video cropping and text/bitmap overlay. Without it, video stream can be much faster.

Possible values are:
vcxNo=0,
vcxBoth=1,
vcxPreviewOnly=2,
vcxGrabOnly=3

"vcxNo" means fastest preview and capture,
"vcxGrabOnly" is little to nothing slower but offer still video frame grabbing capability (with GrabFrame method) without text/image overlay,

"vcxPreviewOnly" can be used when you want to put text/image overlay on video preview while capturing unmodified video,
"vcxBoth" means that both video preview and captured video will have text/image overlays.

VideoCapX.UseVideoFilter = vcxNo
VideoCapX.Connected = True
VideoCapX.Preview = True

Version

Property Version As String

Returns VideoCapX.OCX version number

VideoCodecIndex

Property VideoCodecIndex As Long

Set index of video codec to use for on-the-fly compression of video

VideoCodecQuality

Property VideoCodecQuality As Long

Sets quality parameter for video codec

VideoDeviceIndex

Property VideoDeviceIndex As Long

Set index of video device to use for capture

VideoFlip

Property VideoFlip As Long

Sets video flipping. Flips video image horizontally and/or vertically.

VideoFlip values are:
0 = no flipping (default)
1 = horizontral flip
2 = vertical flip
3 = horizontal+vertical flip

If the video capture device doesn't support video flipping, you can use VideoCapX video filter to manualy flip the image. Just use -1, -2 or -3 as values for this propery.

If VideoFlip property returns -1, it indicates that Connected property should be set to TRUE. If VideoFlip value is -2, it means that video flipping isn't supported by current video device.

VideoCapX.Connected = True
VideoCapX.Preview = True
VideoCapX.VideoFlip = 1

VideoHeight

Property VideoHeight As Long

Returns current video height in pixels. This property is read-only.

VideoInputIndex

Property VideoInputIndex As Long

Sets channel to use on multi-port capture cards

VideoRenderer

Property VideoRenderer As vcxVideoRendererEnum

Select video renderer to use for video preview.

0 = vcxSystemDefault: Use default DirectShow video renderer,
1 = vcxVMR9: Use new Direct3D VMR9 renderer
2 = vcxVMR7: use older VMR7 DirectDraw-based renderer
3 = vcxGDI: use GDI-based video renderer

VideoSourceURL

Property VideoSourceURL As String

URL of network camera acting as video source

VideoDeviceIndex property must be set to -2 for this property to work.

```
vcx.VideoDeviceIndex = -2
vcx.VideoSourceURL = "http://mycam/image.jpg"
vcx.AudioDeviceIndex = -1

vcx.Connected = True
```

vcx.Preview = True

VideoWidth

Property VideoWidth As Long

Returns current video width in pixels. This property is read-only.

WMAttributes

Property WMAttributes As String

Windows Media attributes to set when capturing into WMV files or broadcasting

WMAttributes property is a string delimited with '|' with these fields: Title, Author, Copyright, Rating, Description . It can be an empty string if no attributes are needed.

vcx.WMAttributes="my title|author is me|copyright to me|rating is 5|this is description "

WMTVersion

Property WMTVersion As Long

Sets WindowsMedia system profiles version to use. See ProfileIndex property. Deafult is 7. Possible values are 4,7 and 8.

Methods

AboutBox

Sub AboutBox

Shows About Box of VideoCapX

Acquire

Function Acquire As stdole.Picture

Acquires an image from TWAIN source (like scanner).

Form1.Picture = VideoCapX.Acquire

AllocCapFile

Function AllocCapFile (FileSizeMb As Long) As Long

Pre-allocates space on disk for capture file

You can improve streaming capture performance significantly by preallocating a capture file large enough to store an entire video clip and by defragmenting the capture file before capturing the clip.

AutoTune

Function AutoTune (nChannel As Long) As Long

The AutoTune method scans for a precise signal on the channel's frequency.

TV channels generally map to a unique frequency depending on regional variances. To avoid interference between multiple transmitters that are assigned the same channel when they are in close geographic proximity, small frequency offsets are introduced at each transmitter. In the United States, this offset ranges up to +/– 26.25 kilohertz (kHz).

This method handles the channel-to-frequency conversion and scans for the most precise frequency. Store these values by calling the StoreAutoTune method.
You can find base frequencies for channels in the appendix "International Analog TV Tuning" of DirectShow SDK documentation.

CameraControlGet

Function CameraControlGet (Prop As Long) As Long

Returns value of camera-control properties

Prop values are:
CameraControl_Pan = 0
CameraControl_Tilt = 1
CameraControl_Roll = 2
CameraControl_Zoom = 3
CameraControl_Exposure = 4
CameraControl_Iris = 5
CameraControl_Focus = 6

CameraControl_Pan

Specifies the camera's pan setting, in degrees. Values range from –180 to +180, with the default set to zero. Positive values are clockwise from the origin (the camera rotates clockwise when viewed from above), and negative values are counterclockwise from the origin.

CameraControl_Tilt

Specifies the camera's tilt setting, in degrees. Values range from –180 to +180, with the default set to zero. Positive values point the imaging plane up, and negative values point the imaging plane down.

CameraControl_Roll

Specifies the camera's roll setting, in degrees. Values range from –180 to +180, with the default set to zero. Positive values cause a clockwise rotation of the camera along the image-viewing axis, and negative values cause a counterclockwise rotation of the camera.

CameraControl_Zoom

Specifies the camera's zoom setting, in millimeters. Values range from 10 to 600, and the default is specific to the device.

CameraControl_Exposure

Specifies the exposure setting, in log base 2 seconds. In other words, for values less than zero, the exposure time is 1/2n seconds, and for values zero or above, the exposure time is 2n seconds. For example:

Value Seconds
-3 1/8
-2 1/4
-1 1/2
0 1
1 2
2 4

CameraControl_Iris

Specifies the camera's iris setting, in units of fstop * 10.

CameraControl_Focus

Specifies the camera's focus setting, as the distance to the optimally focused target, in millimeters. The range and default value are specific to the device.

CameraControlGetRange

Function CameraControlGetRange (Prop As Long, MinVal As Long, MaxVal As Long, SteppingDelta As Long, DefaultValue As Long, CapsFlags As Long) As Long

Retrieve camera control range values.

Prop values are:
CameraControl_Pan = 0
CameraControl_Tilt = 1
CameraControl_Roll = 2
CameraControl_Zoom = 3
CameraControl_Exposure = 4
CameraControl_Iris = 5
CameraControl_Focus = 6

Return value 0 means success, otherwise, it's one of DirectX error codes.

CapsFlags arguments receive value of 1 if the property is controlled automatically, or 2 if it is controlled manualy, or 3 if both is possible.

CameraControlSet

Function CameraControlSet (Prop As Long, Val As Long) As Long

Sets camera-control properties

Prop values are:
CameraControl_Pan = 0
CameraControl_Tilt = 1
CameraControl_Roll = 2
CameraControl_Zoom = 3
CameraControl_Exposure = 4
CameraControl_Iris = 5
CameraControl_Focus = 6
CameraControl_Pan

Add 10 to Prop value to use RELATIVE flag.

Specifies the camera's pan setting, in degrees. Values range from –180 to +180, with the default set to zero. Positive values are clockwise from the origin (the camera rotates clockwise when viewed from above), and negative values are counterclockwise from the origin.

CameraControl_Tilt

Specifies the camera's tilt setting, in degrees. Values range from –180 to +180, with the default set to zero. Positive values point the imaging plane up, and negative values point the imaging plane down.

CameraControl_Roll

Specifies the camera's roll setting, in degrees. Values range from –180 to +180, with the default set to zero. Positive values cause a clockwise rotation of the camera along the image-viewing axis, and negative values cause a counterclockwise rotation of the camera.

CameraControl_Zoom

Specifies the camera's zoom setting, in millimeters. Values range from 10 to 600, and the default is specific to the device.

CameraControl_Exposure

Specifies the exposure setting, in log base 2 seconds. In other words, for values less than zero, the exposure time is 1/2n seconds, and for values zero or above, the exposure time is 2n seconds. For example:

Value Seconds
-3 1/8
-2 1/4
-1 1/2
0 1
1 2
2 4

CameraControl_Iris

Specifies the camera's iris setting, in units of fstop * 10.

CameraControl_Focus

Specifies the camera's focus setting, as the distance to the optimally focused target, in millimeters. The range and default value are specific to the device.

CompareImages

Function CompareImages (Picture1 As stdole.Picture, Picture2 As stdole.Picture, Sensitivity As Long) As Long

Returns difference between two images.

Suggested value for Sensitivity parameter is 25.
This method returns percent of different pixels in two images. So, return value is 0 for identical images and 100 for completly different.

Picture1.Picture = VideoCapX.GrabFrame
-or-
Picture1.Picture = VideoCapX.ReceiveFrame (serveraddr)

```
If VideoCapX.CompareImages(Picture1, Picture2, 25) > 20 Then
   MsgBox "Where are you going?"
End If
```

ConnectionState

Function ConnectionState As Long

returns current DisplayRemote connection state

0 = no connection
1 = resolving network address
2 = connecting
3 = connected to host
4 = connection established

CopyCaptureFile

Function CopyCaptureFile (New As String) As Long

Copies AVI file from pre-allocated storage into new file

CopyFrame

Function CopyFrame As Boolean

Copy current vide frame into clipboard

none

DetectMotion

Function DetectMotion (Sensitivity As Long) As Long

Detect changes in video frames

The DetectMotion method returns the number between 0 and 100, reflecting the change detected in front of the camera.

Call this method every second or two and check the result it returns. If the result is greater than 30, there is something moving in front of the camera. Experiment with this to see which value to use.

See SetMotionMask method.

```
Sub Timer1_OnTimer()
If VideoCapX1.DetectMotion>30 then
Beep
MsgBox “Where are you going?”
End If
End Sub
```

DetectObjects

Function DetectObjects (Threshold As Long) As String

Detects moving objects on the image and returns their coordinates

DisplayRemote

Function DisplayRemote (RemoteAddress As String, Audio As Boolean) As Long

Starts a video-cenference call

```
'call a computer with IP address of 192.168.0.5
clientvcx.DisplayRemote "192.168.0.5", True

'or, with a network address as 'johnsoffice'
clientvcx.DisplayRemote "johnsoffice", True
```

ExportToDV

Function ExportToDV (filename As String) As Long

Export DV-compatible AVI file to DV VTR device

First, connect to DV device. Then call ExportToDV method.
Use VCRSetMode to start RECORD mode on DV VTR and STOP it after the file is finished playing.
Use PlayerGetPos to track progress. PlayerEnd event is triggered on export finish.
After you finish exporting to DV, call PlayerClose.
If you abort this method (by exiting your application), don't forget to STOP video recorder (using VCRSetMode method).

```
VideoCapX.Connected=True
VideoCapX.ExportToDV "movie.avi"
```

GetActualFrameRate

Function GetActualFrameRate As Double

Returns current actual frame rate. Some devices may provide lower frame rates than requested because of bandwidth availability. This is only available during video streaming.

GetAudioCodecCount

Function GetAudioCodecCount As Long

Returns installed audio codec count

GetAudioCodecName

Function GetAudioCodecName (Index As Long) As String

Returns audio codec name

GetAudioDeviceCount

Function GetAudioDeviceCount As Long

Returns number of audio devices in the system

GetAudioDeviceName

Function GetAudioDeviceName (Index As Long) As String

Returns audio device name

GetAudioFormat

Function GetAudioFormat (FmtTag As Long, nChannels As Long, nSamplesPerSec As Long, nAvgBytesPerSec As Long, nBlockAlign As Long, wBitsPerSample As Long) As Boolean

Returns audio format parameters

GetAudioInputCount

Function GetAudioInputCount As Long

Returns number of input ports on audio source

GetAudioInputName

Function GetAudioInputName (Index As Long) As String

Returns audio input port name

GetAudioLevel

Function GetAudioLevel As Long

Returns audio level in preview mode

GetAudioLevel2

Function GetAudioLevel2 (left As Long, right As Long) As Long

Get audio levels for left and right channel

GetCapFileSize

Function GetCapFileSize As Double

Returns file size (in bytes) of capture file

On error, value of -1 is returned.
By using this method, you can check capture file size while capturing is in progress. That way, it gives you a chance to stop capture, change CapFilename and Startcapture again, if you want to keep capture files under some size limit.

GetCapInfo

Function GetCapInfo (ValueIndex As Long) As String

method GetCapInfo

GetCapStatus

Function GetCapStatus (ImageWidth As Long, ImageHeight As Long, CurrentVideoFrame As Long, CurrentVideoFramesDropped As Long, CurrentTimeElapsedMS As Long, fCapturingNow As Long) As Long

Retrieves video capture parameters

Returns TRUE if successful or FALSE if window is not connected to a capture driver.

This method returns various status information in variables passed to it as arguments. ImageHeihgt and Width are in pixels, fCapturingNow is TRUE is capture is in progress.

GetControlRef

Function GetControlRef As Long

Get control reference pointer

GetDateCode

Function GetDateCode As Date

Returns time on DV video tape (the time when the video has been taken)

See GetTimecode method.

GetDeviceDesc

Function GetDeviceDesc (DeviceIndex As Long) As String

Returns device description string

GetDeviceID

Function GetDeviceID (DeviceIndex As Long) As String

Returns unique device ID string

If DeviceIndex parameter is in range 0-1000, video device ID is returned. If DeviceIndex is in range 1000-2000, audio device ID is returned (first audio device is 1000, second 1001, etc)

GetFilterSettings

Function GetFilterSettings (FilterID As Long) As String

Returns current filter settings

FitlerID values:
1 = video compress filter
2 = audio compress filter
3 = video source filter
4 = audio source filter
5 = user filter
This method returns current settings of the filter as string. To set filter, call SetFilterSettings method.

If returned value is an empty string, selected filter doesn't support settings retrieval by code.

```
Dim s As String
VideoCapX.VideoCodecIndex = 6
VideoCapX.ShowVideoCodecDlg
s=VideoCapX.GetFilterSettings(1)
```

GetFrameAsHBITMAP

Function GetFrameAsHBITMAP As Long

Return Windows HBITMAP value of current video frame.

.

GetProfileCount

Function GetProfileCount As Long

Returns number of WindowsMedia system profiles

See WMTVersion property.

GetProfileDesc

Function GetProfileDesc (ProfileIndex As Long) As String

Returns WindowsMedia profile description

GetProfileName

Function GetProfileName (ProfileIndex As Long) As String

Returns WindowsMedia profile name

GetRGB

Function GetRGB

Returns current video frame image as array of RGB values.

.

```
Dim a() As Byte
a = VideoCapX1.GetRGB
For y = 0 To 239 ‘image height is 240 pixels in this case
For x = 0 To 319 ‘320 pixels
i = (y * (320 * 3)) + (x * 3)
'NOTE: byte order isn't RGB, it's BGR
PSet (x, y), RGB(a(i + 2), a(i + 1), a(i))
Next x
Next y
```

GetTimecode

Function GetTimecode As Long

Return timecode value on digital VCR video type

Returned value is:
Hours, minutes, seconds, and frames, as a binary coded decimal (BCD) value: 0xhhmmssff.

```
Dim h As Long, m As Long, s As Long, f As Long, tc As Long, str As String
tc = vcx.GetTimecode
str = String(8 - Len(Hex(tc)), "0") + Hex(tc)
s = Val(Mid(str, 5, 2))
m = Val(Mid(str, 3, 2))
h = Val(Mid(str, 1, 2))
f = Val(Mid(str, 7, 2))
```

Label1.Caption = h & ":" & m & ":" & s & ":" & f

GetTunerSignal

Function GetTunerSignal As Long

Returns 1 if TV signal is present on current channel

GetVideoCaps

Function GetVideoCaps

Returns an array of supported video formats

Call this method after Connected has been set to TRUE, but, before starting video preview.
If device drivers doesn't support this feature, this method returns empty variable.

Returned array has four columns:
1. Video width
2. Video height
3. Bit per pixel value
4. Color format (see ColorFormat property for list of values)

```
Dim a, f
a = vcx.GetVideoCaps()
vidsize.Clear
For f = LBound(a) To UBound(a)
vidsize.AddItem a(f, 0) & "x" & a(f, 1) & "x" & a(f, 2) & "," & a(f, 3)
Next f
```

GetVideoCodecCount

Function GetVideoCodecCount As Long

Returns installed video codec count

GetVideoCodecName

Function GetVideoCodecName (nIndex As Long) As String

Returns video codec name.

GetVideoDeviceCount

Function GetVideoDeviceCount As Long

Returns number of video-capture devices in the system

GetVideoDeviceDesc

Function GetVideoDeviceDesc (Index As Long) As String

Returns video device description

GetVideoDeviceName

Function GetVideoDeviceName (Index As Long) As String

Returns video device name

Index parameter is in range 0 to GetVideoInputCount-1

GetVideoFormat

Function GetVideoFormat (width As Long, height As Long) As Long

Returns video size.

Returns TRUE if successful or FALSE if control is not connected to a capture driver.

ImageHeight and ImageWidth are in pixels.

```
Dim w As Long, h As Long
VideoCapX.GetVideoFormat w,h
```

GetVideoInputCount

Function GetVideoInputCount As Long

Returns number of video inputs on currently selected video device (card).

Returns 0 if your video device doesn't have multiple video inputs.

GetVideoInputName

Function GetVideoInputName (Index As Long) As String

Returns name of specified video channel on multiple-input capture cards.

GetVideoProcAmp

Function GetVideoProcAmp (ValueIndex As Long) As Long

Get video properties

GetVideoProcAmpValueRange

Function GetVideoProcAmpValueRange (ValueIndex As Long, Min As Long, Max As Long, SteppingDelta As Long, Default As Long) As Long

Retrieve value range for video property.

ValueIndex parameter is:
Brightness = 0,
Contrast = 1,
Hue = 2,
Saturation = 3,
Sharpness = 4,
Gamma = 5,
ColorEnable = 6,
WhiteBalance = 7,
BacklightCompensation = 8,
Gain = 9

```
Dim Brightness As Long

VideoCapX.GetVideoProcAmp 0, Brightness

Dim MinVal As Long ,MaxVal As Long ,StepD As Long ,DefVal As Long
VideoCapX.GetVideoProcAmpValueRange 0, MinVal, MaxVal, StepD, DefVal

'set brightness here
Brightness=MinVal+((MaxVal-MinVal)/2)
```

VideoCapX.SetVideoProcAmp 0, Brightness

GetVRIUnknown

Function GetVRIUnknown As Unknown

Returns IUnknown interface of current video renderer filter. See 'VideoRenderer' property.

GrabFrame

Function GrabFrame As stdole.Picture

Returns current video frame as VB Picture object

With this method you can load video frame into PictureBox control.

PictureBox.Picture = VideoCapX.GrabFrame

HTTPUpload

Function HTTPUpload (WebServer As String, WebPage As String, Fields As String, Files As String)

Use HTTP upload protocol to send information and files to web server

Arguments:
WebServer = web server address
WebPage = name of upload web page
Fields = list of 'fieldname' and 'fieldvalue' values delimited with '|'
Files = list of 'fieldname' and 'file path' values delimited with '|'

Returns TRUE if successful, or FALSE otherwise.

vcx.HTTPUpload "www.mysite.com" , "upload.asp",
"field1|value1|field2|value2" ,
"file1|c:\folder\mypic.jpg|file2|c:\folder\myvideo.avi"

LoadProfileFromURL

Function LoadProfileFromURL (URL As String) As Long

Loads WM profile data from .prx file. URL argument must start with 'http://' or 'file://' . Return value is 1 on success or 0 on failure.

PauseCapture

Function PauseCapture As Long

Pause capture

Ping

Function Ping (ServerAddress As String, Timeout As Long) As Long

method Ping

PlayerClose

Function PlayerClose As Long

Closes media playback.

PlayerGetFrame

Function PlayerGetFrame As Double

Returns current video frame number

See UseVideoFilter.

PlayerGetFrameCount

Function PlayerGetFrameCount As Double

Returns number of fideo frames in the movie

PlayerGetLenMS

Function PlayerGetLenMS As Long

Returns media file length in milliseconds.

PlayerGetPos

Function PlayerGetPos As Long

Returns current playing position.

PlayerGetVideoSize

Function PlayerGetVideoSize (width As Long, height As Long) As Long

Returns video size for media player.

PlayerOpen

Function PlayerOpen (filename) As Long

Opens media file for playback.

PlayerOpenDVD

Function PlayerOpenDVD (path As String, AudioStream As Long, SubpictureStream As Long) As Long

Open DVD volume

PlayerSetFrame

Function PlayerSetFrame (nFrame As Double) As Double

Sets current video frame

PlayerSetFullScreen

Function PlayerSetFullScreen (Full As Long) As Long

Opens full-screen playback.

PlayerSetMute

Function PlayerSetMute (Mute As Long) As Long

Mutes sound of media player.

PlayerSetPos

Function PlayerSetPos (PosMS As Double) As Long

Sets current position for playback.

PlayerSetRate

Function PlayerSetRate (NewRate As Double) As Double

Sets the playback rate

The playback rate is expressed as a ratio of the normal speed. Thus, 1.0 is normal playback speed, 0.5 is half speed, and 2.0 is twice speed. For audio streams, changing the rate also changes the pitch.

Negative values indicate reverse playback. Most filters do not support negative playback, but instead return an error code if the dRate parameter is negative.

If NewRate parameter is 0, this method returns current playback rate.
Call this method only after player is initialized with PlayerOpen method.

PlayerSetSize

Function PlayerSetSize (width As Long, height As Long) As Long

Sets media player window size.

PlayerStart

Function PlayerStart As Long

Start playing media file.

PlayerStepFrames

Function PlayerStepFrames (nFrames As Long) As Long

Steps nFrames in player mode

PlayerStepOneFrame

Function PlayerStepOneFrame As Long

Display next frame

PlayerStop

Function PlayerStop As Long

Stop playing media file.

PlayRemoteAudio

Function PlayRemoteAudio (RemoteAddress As String) As Long

Connects to remote VideoCapX server and receives only audio stream.

ReceiveAudio

Function ReceiveAudio (ServerAddress As String, Play As Boolean, nChannels As Long, nSamplesPerSecond As Long, nBytesPerSample As Long, PCMData) As Long

Receive audio data packet from server

If you are not interested in PCM data, set last four parameters to 0.
To play a received sound, set Play parameter to TRUE.
Address parameter is a network address of remote VideoCapX server.

VideoCapX.ReceiveAudio "127.0.0.1", True, 0, 0, 0, 0

ReceiveFrame

Function ReceiveFrame (ServerName As String) As stdole.Picture

Return video frame from remote server as Picture object.

Remote server must run VideoCapX control with ServerMode set to TRUE.
Also, ServerPort and ServerPAssword on both computers must be set to identical value.

Picture1.Picture = VideoCapX.ReceiveFrame("myvideoserver")

Recompress

Function Recompress (SrcFile As String, DestFile As String) As Long

Copies AVI/WMV into new file using specified video compression.

If DestFile has an .AVI extension, VideoCodecIndex/AudioCodecIndex settings are used, or, if DestFile has .WMV extension, ProfileIndex/ProfileData settings are used for compression.

RecompressEx

Function RecompressEx (SrcFile1 As String, SrcFile2 As String, DestFile As String, TimeStart As Double, TimeEnd As Double) As Long

Use this method to merge video and audio files and/or crop video files.

SrcFile1 is source video.
SrcFile2 is source audio MP3, WAV, WMA or AVI file.
DestFile is file to be created.
TimeStart and TimeEnd are time boundaries in milliseconds. If cropping is not needed, set these arguments to 0.
Destination AVI (or WMV) file is compressed using currently selected video/audio codec (or WM profile) .

Cut video file:
RecomrpessEx "original.avi","","new.avi",5000,15000

Merge video and audio into new file.
RecompressEx "video.avi", "audio.wav", "new.avi", 0, 0

ResumeCapture

Function ResumeCapture As Long

Resume capture paused with PauseCapture method

SaveBuffer

Sub SaveBuffer (fromTime As Double, timeLength As Double)

Saves current video buffer into a file with a name specified by CapFilename property and using video/audio codecs specified with VideoCodecIndex/AudioCodecIndex.

"fromTime" and "timeLength" parameters are in milliseconds from the beginning of the buffer.
To save 5 seconds of video starting at 8-th second in buffer, use SaveBuffer(8000,5000) .

While the operation is running, RecompressProgress event is fired after every video frame saved.

SaveFrame

Function SaveFrame (filename As String) As Boolean

Saves video image into file

SaveFrameJPG

Function SaveFrameJPG (filename As String, quality As Long) As Boolean

Saves current video frame into JPG file

Quality parameter is a JPEG image quality setting (0-100).

Vcx1.SaveFrameJPG("C:\mypic.jpg",80)

SavePictureJPG

Function SavePictureJPG (Picture As stdole.Picture, filename As String, quality As Long) As Long

Saves Picture object into JPG file.

Quality parameter is a JPEG image quality setting (0-100).

If GetHBitmap() method of .NET Bitmap class is used to get bitmap handle, it must be deleted after use to avoid memory leaks.

Example .NET code:

```
[System.Runtime.InteropServices.DllImport("gdi32.dll")]
public static extern bool DeleteObject(IntPtr hObject);

private void axVideoCapX1_NewFrame(object sender, EventArgs e)
{
pictureBox1.Image = axVideoCapX1.GrabFrame();
{
IntPtr hb = ((System.Drawing.Bitmap)(pictureBox1.Image)).GetHbitmap();
axVideoCapX2.SingleFrameAddPicture(hb.ToInt32());
DeleteObject(hb);
}
}
```

```
Vcx1.SaveFrameJPG(Picture1.Picture,“C:\mypic.jpg”,80)
Vcx1.SaveFrameJPG(LoadPicture(“c:\windows\setup.bmp”),“C:\setup.jpg”,90)
```

SelectSource

Function SelectSource As Long

Let the user select TWAIN source.

SendScriptCommand

Function SendScriptCommand (Type As String, Data As String) As Long

Send script type/command pair to broadcast client. This method works only if WM broadcast started with StartBroadcast method is running.

Before you call StartBroadcast, you must load a custom WM profile with script stream enabled. See ProfileData property. To create a custom WM profile, you can use Windows Media Profile Editor tool available for free at Microsoft's web site. If currently selected WM profile doesn't have script stream configured, this method returns -1. On success, this method returns 0.

The following table lists script types that are supported by Windows Media Player.

Script type	Description
URL	The player sends the specified URL to the browser for display to the user. If an embedded player control is being used, you can add a specific frame reference to the URL by using the &&*framename* syntax.
FILENAME	A URL to another media file to be played.
CAPTION	A text string that is displayed in the captions area of Windows Media Player. This type supports standard HTML formatting, so the text can be formatted as you wish. An example of use is closed captioning.
EVENT	The name of an event that is to occur. The EVENT type supports customization for your own uses. The code for the specified event must be defined in the Windows Media metafile for the stream in order for the player to perform the specified event. An example of use is ad insertion.
OPENEVENT	This script precedes the actual EVENT. The OPENEVENT allows the player to pre-buffer the content so that when the EVENT occurs, the switch between streams appears to be seamless.
TEXT	A TEXT string that is displayed in the captions area of Windows Media Player. Can be plain text, SAMI, or HTML formatted text.

vcx.SendScriptCommand "TEXT", "this is a caption / subtitle"

SetAudioBalance

Function SetAudioBalance (Balance As Long) As Long

Sets the balance of the audio signal

SetAudioDelay

Function SetAudioDelay (DelayMS As Long) As Long

Sets audio delay (positive or negative) in captured AVI file. DelayMS argument is in milliseconds.

SetAudioFormat

Function SetAudioFormat (FmtTag As Long, nChannels As Long, nSamplesPerSec As Long, nAvgBytesPerSec As Long, nBlockAlign As Long, nBitsPerSample As Long) As Boolean

Sets audio format for capture

FormatTag

Waveform-audio format type. Format tags are registered with Microsoft Corporation for many compression algorithms. A complete list of format tags can be found in the MMREG.H header file.
WAVE_FORMAT_PCM = 1

nChannels

Number of channels in the waveform-audio data. Monaural data uses one channel and stereo data uses two channels.

nSamplesPerSec

Sample rate, in samples per second (hertz), that each channel should be played or recorded. If wFormatTag is WAVE_FORMAT_PCM, then common values for nSamplesPerSec are 8.0 kHz, 11.025 kHz, 22.05 kHz, and 44.1 kHz. For non-PCM formats, this member must be computed according to the manufacturer's specification of the format tag.

nAvgBytesPerSec

Required average data-transfer rate, in bytes per second, for the format tag. If wFormatTag is WAVE_FORMAT_PCM, nAvgBytesPerSec should be equal to the product of nSamplesPerSec and nBlockAlign. For non-PCM formats, this member must be computed according to the manufacturer's specification of the format tag.

Playback and record software can estimate buffer sizes by using the nAvgBytesPerSec member.

nBlockAlign

Block alignment, in bytes. The block alignment is the minimum atomic unit of data for the wFormatTag format type. If wFormatTag is WAVE_FORMAT_PCM, nBlockAlign should be

equal to the product of nChannels and wBitsPerSample divided by 8 (bits per byte). For non-PCM formats, this member must be computed according to the manufacturer's specification of the format tag.

Playback and record software must process a multiple of nBlockAlign bytes of data at a time. Data written and read from a device must always start at the beginning of a block. For example, it is illegal to start playback of PCM data in the middle of a sample (that is, on a non-block-aligned boundary).

nBitsPerSample

Bits per sample for the wFormatTag format type. If wFormatTag is WAVE_FORMAT_PCM, then wBitsPerSample should be equal to 8 or 16. For non-PCM formats, this member must be set according to the manufacturer's specification of the format tag. Note that some compression schemes cannot define a value for wBitsPerSample, so this member can be zero.

SetAudioInputLevel

Function SetAudioInputLevel (Level As Long) As Long

Sets the recording level for audio input selected with AudioInputIndex property. Level value is in range 0 to 100.

Call this method after you have set Connected property to TRUE and after you set AudioInputIndex property.

SetAudioVolume

Function SetAudioVolume (Volume As Long) As Long

Sets audio renderer volume. Volume argument range is 0 - 100. if Volume is -1, this method returns current audio volume.

SetBitmapOverlay

Function SetBitmapOverlay (Index As Long, BitmapHandle As Long, left As Long, top As Long, width As Long, height As Long, TransColor As Long, Alpha As Long) As Long

Sets bitmap to show on-video

There can be 40 bitmap overlays on video, use Index parameter (range 0-39) to set overlays.

BitmapHandle parameter is a Windows handle of the bitmap.
For standard PictureBox control, use Picture.Handle property to get this value.

TransColor parameter is a RGB value of transparent color, if no transparency is used, set this parameter to -1.
Alpha paramter is in range 0 (transparent) to 255 (opaque) .

Use SetBitmapOverlay 0,0,0,0,0 to remove bitmap overlay.

VideoCapX.SetBitmapOverlay 0, Picture1.Picture.Handle, 0, 0, 320,240, -1,127

SetChromaKey

Function SetChromaKey (BackImage As String, MinTransparentColor As Long, MaxTransparentColor As Long, BackOrFront As vcxCKFlags) As Long

Sets chrom-key effect. BackImage parameter is a filename of static image background which will be visible through video. Color paramters are RGB values.

Parameter:

BackOrFront=1 ... Back image is behind video
BackOrFront=0 ... Back image is in front of video

MinTransparentColor=-2 ... "Green key" effect. Green color is replaced.
MinTransparentColor=-3 ... "Blue key" effect. Blue color is replaced.

In case green or blue key is used, MaxTransparentColor specifies tolerance.

```
' view lake image through dark pixels on video
vcx.SetChromaKey App.Path + "\lake.jpg", 0, RGB(90, 90, 90), 1
```

'view video through green pixels in effect.avi video .
vcx.SetChromaKey "c:\effect.avi", -2, 20, 0

SetCrop

Function SetCrop (x As Long, y As Long, W As Long, H As Long) As Long

Crops live-video stream to the rectangle of dimensions (W,H) and with top-left coordinate of (X,Y).

This method must be called before connecting VideoCapX to video source.

To capture a face in front of camera (if camera has 320x240 resolution), use:

VideoCapX.SetCrop 110, 45, 100, 150

SetFadeLevel

Function SetFadeLevel (NewLevel As Long) As Long

Sets fade level (the range is 0-100, 0 is neutral) for video

SetFilterSettings

Function SetFilterSettings (Filter As Long, Data As String) As Long

Set filter settings

Fitler argument values:
1 = video compress filter
2 = audio compress filter
3 = video source filter
4 = audio source filter
5 = user filter

This method returns 0 on success or negative value on error.
Data parameter is string retrieved by GetFitlerSettings method. Make sure that same filter is selected as when GetFilterSettings is called. Every fitler returns/accepts different format of settings binary data.

```
'before this, load filter data in string variable d
VideoCapX.VideoCodecIndex = 6
VideoCapX.SetFitlerSettings 1, d
```

SetHighPriority

Function SetHighPriority (High As Boolean) As Boolean

Sets priority class for the current process. High argument is TRUE for high priority or FALSE for normal priority.

SetMasterAudioVolume

Function SetMasterAudioVolume (LineID As Long, Volume As Long) As Long

Set master volume for specified mixer line. Volume argument range is 0 - 100. if Volume is -1, this method returns current audio volume.

LineID values:

DST_DIGITAL	1
DST_LINE	2
DST_MONITOR	3
DST_SPEAKERS	4
DST_HEADPHONES	5
DST_TELEPHONE	6
DST_WAVEIN	7
DST_VOICEIN	8
SRC_DIGITAL	11

SRC_LINE	12
SRC_MICROPHONE	13
SRC_SYNTHESIZER	14
SRC_COMPACTDISC	15
SRC_TELEPHONE	16
SRC_PCSPEAKER	17
SRC_WAVEOUT	18
SRC_AUXILIARY	19
SRC_ANALOG	20

SetMotionMask

Function SetMotionMask (Index As Long, left As Long, top As Long, width As Long, height As Long) As Long

Sets rectangle(s) to ignore on the image while detecting motion.

Index parameter is in range 0 - 9. Dimmensions are in pixels.

SetOption

Function SetOption (OptionIndex As Long, Value)

method SetOption

SetSlaveControl

Function SetSlaveControl (Index As Long, CtlRef As Long, PosX As Long, PosY As Long, PosW As Long, PosH As Long, BorderWidth As Long, BorderColor As Long) As Long

Set video source control

SetTextOverlay

Function SetTextOverlay (Index As Long, Caption As String, x As Long, y As Long, FontName As String, FontSize As Long, FColor As Long, BColor As Long) As Long

Sets on-video text caption.

Returns TRUE on succes, FALSE otherwise.

ID parameter is in range 0 to 19.
X, Y and FontSize parameters are in device pixels.
Use VisualBasic RGB function to set TextColor and TextBgColor parameters.

Set TextBGColor parameter to –1 for transparent text output.
Special values are “TIME” for Text parameter to show time-stamp and "SMPTE" for SMPTE-format time display.
To clear text caption, use SetTextOverlay method with an empty string as Text parameter.

To set date-time stamp in top-left corner on video, use:

VideoCapX.SetTextOverlay 0, ”TIME”, 0, 0, ”Arial”, 14, RGB(255,0,0), -1

SetTunerChannel

Function SetTunerChannel (nChannel As Long, VideoSubChannel As Long, AudioSubChannel As Long) As Long

Set TV tuner channel

If nChannel argument is -1, the method returns current channel number.

If nChannel argument is -2, the method returns minimal channel number.
If nChannel argument is -3, the method returns maximal channel number.

Error return values:
-1 = Connected property not set
-2 = No TV tuner detected
-3 = Set/get channel failed

SetTunerCountryCode

Function SetTunerCountryCode (Code As Long) As Long

Set/Get TV tuner country code

Codes are:

// Each entry consists of:
// Country/region code
// Cable frequency table ID
// Broadcast frequency table ID
// Analog video standard
1, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // USA
// Anguilla
// Antigua
// Bahamas
// Barbados
// Bermuda
// British Virgin Islands
// Canada
// Cayman Islands
// Dominica
// Dominican Republic
// Grenada
// Jamaica
// Montserrat
// Nevis
// St. Kitts
// St. Vincent and the Grenadines
// Trinidad and Tobago
// Turks and Caicos Islands
// Barbuda
// Puerto Rico
// Saint Lucia

// United States Virgin Islands
2, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Canada
20, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Egypt
212, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_SECAM_B, // Morocco
213, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Algeria
216, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Tunisia
218, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Libya
220, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Gambia
221, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Senegal Republic
222, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_SECAM_B, // Mauritania
223, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_K, // Mali
224, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_K, // Guinea
225, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_SECAM_K, // Cote D'Ivoire
226, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_K, // Burkina Faso
227, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Niger
228, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Togo
229, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Benin
230, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Mauritius
231, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Liberia
232, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Sierra Leone
233, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Ghana
234, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Nigeria
235, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Chad
236, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Central African Republic
237, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Cameroon
238, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_NTSC_M, // Cape Verde Islands
239, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Sao Tome and Principe
240, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_SECAM_B, // Equatorial Guinea
241, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Gabon
242, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_D, // Congo
243, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Congo(DRC)
244, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_PAL_I, // Angola
245, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_NTSC_M, // Guinea-Bissau
246, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Diego Garcia
247, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Ascension Island
248, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Seychelle Islands
249, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Sudan
250, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B,, // Rwanda
251, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Ethiopia
252, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Somalia
253, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Djibouti
254, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Kenya
255, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Tanzania
256, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Uganda
257, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_SECAM_K, // Burundi
258, F_UNI_CABLE, F_FIX_BROAD, AnalogVideo_PAL_B, // Mozambique
260, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Zambia

261, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Madagascar
262, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Reunion Island
263, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Zimbabwe
264, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_I, // Namibia
265, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Malawi
266, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_I, // Lesotho
267, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_K, // Botswana
268, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Swaziland
269, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_K, // Mayotte Island
269, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Comoros
27, F_UNI_CABLE, F_UK__BROAD, AnalogVideo_PAL_I, // South Africa
290, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // St. Helena
291, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Eritrea
297, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Aruba
298, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Faroe Islands
299, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Greenland
30, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Greece
31, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Netherlands
32, F_WEU_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Belgium
33, F_UNI_CABLE, F_FRA_BROAD, AnalogVideo_SECAM_L, // France
34, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Spain
350, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Gibraltar
351, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Portugal
352, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Luxembourg
353, F_UNI_CABLE, F_IRE_BROAD, AnalogVideo_PAL_I, // Ireland
354, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Iceland
355, F_UNI_CABLE, F_ITA_BROAD, AnalogVideo_PAL_B, // Albania
356, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Malta
357, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Cyprus
358, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Finland
359, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Bulgaria
36, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Hungary
370, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Lithuania
371, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_D, // Latvia
372, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Estonia
373, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Moldova
374, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Armenia
375, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Belarus
376, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Andorra
377, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_G, // Monaco
378, F_UNI_CABLE, F_ITA_BROAD, AnalogVideo_PAL_B, // San Marino
380, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Ukraine
381, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Serbia and Montenegro
385, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Croatia
386, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Slovenia
387, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Bosnia and Herzegovina
389, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // F.Y.R.O.M. (Former

Yugoslav Republic of Macedonia)
39, F_UNI_CABLE, F_ITA_BROAD, AnalogVideo_PAL_B, // Italy
39, F_UNI_CABLE, F_ITA_BROAD, AnalogVideo_PAL_B, // Vatican City
40, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_PAL_D, // Romania
41, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Switzerland
41, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Liechtenstein
420, F_UNI_CABLE, F_CZE_BROAD, AnalogVideo_PAL_D, // Czech Republic
421, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Slovak Republic
43, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Austria
44, F_UK__CABLE, F_UK__BROAD, AnalogVideo_PAL_I, // United Kingdom
45, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Denmark
46, F_WEU_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Sweden
47, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Norway
48, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_PAL_B, // Poland
49, F_WEU_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Germany
500, F_UNI_CABLE, F_UK__BROAD, AnalogVideo_PAL_I, // Falkland Islands (Islas Malvinas)
501, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Belize
502, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Guatemala
503, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // El Salvador
504, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Honduras
505, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Nicaragua
506, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Costa Rica
507, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Panama
508, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // St. Pierre and Miquelon
509, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Haiti
51, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Peru
52, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Mexico
53, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Cuba
53, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Guantanamo Bay
54, F_USA_CABLE, F_USA_BROAD, AnalogVideo_PAL_N, // Argentina
55, F_USA_CABLE, F_USA_BROAD, AnalogVideo_PAL_M, // Brazil
56, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Chile
57, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Colombia
58, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Bolivarian Republic of Venezuela
590, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Guadeloupe
590, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // French Antilles
591, F_USA_CABLE, F_USA_BROAD, AnalogVideo_PAL_N, // Bolivia
592, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Guyana
593, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Ecuador
594, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // French Guiana
595, F_USA_CABLE, F_USA_BROAD, AnalogVideo_PAL_N, // Paraguay
596, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // Martinique
597, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Suriname
598, F_USA_CABLE, F_USA_BROAD, AnalogVideo_PAL_N, // Uruguay
599, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Netherlands Antilles

60, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Malaysia
61, F_UNI_CABLE, F_OZ__BROAD, AnalogVideo_PAL_B, // Australia
61, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Cocos-Keeling Islands
62, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Indonesia
63, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Philippines
64, F_UNI_CABLE, F_NZ__BROAD, AnalogVideo_PAL_B, // New Zealand
65, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Singapore
66, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Thailand
670, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Saipan Island
670, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Rota Island
670, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Tinian Island
671, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Guam
672, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Christmas Island
672, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Australian Antarctic Territory
672, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Norfolk Island
673, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Brunei
674, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Nauru
675, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Papua New Guinea
676, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Tonga
677, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Solomon Islands
678, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Vanuatu
679, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Fiji Islands
680, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Palau
681, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_SECAM_K, // Wallis and Futuna Islands
682, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Cook Islands
683, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Niue
684, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Territory of American Samoa
685, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Samoa
686, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Kiribati Republic
687, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // New Caledonia
688, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Tuvalu
689, F_UNI_CABLE, F_FOT_BROAD, AnalogVideo_SECAM_K, // French Polynesia
690, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Tokelau
691, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Micronesia
692, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Marshall Islands
7, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Russia
7, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Kazakhstan
7, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Kyrgyzstan
7, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Tajikistan
7, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Turkmenistan
7, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Uzbekistan
81, F_JAP_CABLE, F_JAP_BROAD, AnalogVideo_NTSC_M_J, // Japan
82, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Korea (South)
84, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Vietnam
850, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Korea (North)

852, F_UNI_CABLE, F_UK__BROAD, AnalogVideo_PAL_I, // Hong Kong SAR
853, F_UNI_CABLE, F_UK__BROAD, AnalogVideo_PAL_I, // Macao SAR
855, F_USA_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Cambodia
856, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Laos
86, F_CHN_CABLE, F_CHN_BROAD, AnalogVideo_PAL_D, // China
871, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // INMARSAT (Atlantic-East)
872, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // INMARSAT (Pacific)
873, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // INMARSAT (Indian)
874, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // INMARSAT (Atlantic-West)
880, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Bangladesh
886, F_USA_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Taiwan
90, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Turkey
91, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // India
92, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Pakistan
93, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Afghanistan
94, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Sri Lanka
95, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Myanmar
960, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Maldives
961, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Lebanon
962, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Jordan
963, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Syria
964, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Iraq
965, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Kuwait
966, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Saudi Arabia
967, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Yemen
968, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Oman
971, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // United Arab Emirates
972, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Israel
973, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Bahrain
974, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_PAL_B, // Qatar
975, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_NTSC_M, // Bhutan
976, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Mongolia
977, F_UNI_CABLE, F_USA_BROAD, AnalogVideo_PAL_B, // Nepal
98, F_UNI_CABLE, F_WEU_BROAD, AnalogVideo_SECAM_B, // Iran
994, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Azerbaijan
995, F_UNI_CABLE, F_EEU_BROAD, AnalogVideo_SECAM_D, // Georgia
0, 0, 0, 0 // End of the list

SetTunerInputType

Function SetTunerInputType (InputType As Long) As Long

Set / retrieve TV tuner input type; cable or antenna.

InputType parameter values:
1 = Cable
2 = Antenna
-1 = retrieve current input type

Negative return values indicate an error.

SetTunerMode

Function SetTunerMode (Mode As Long) As Long

Sets a multifunction tuner to the specified mode.

Available mode constants are:
AMTUNER_MODE_DEFAULT = 0x0000,
AMTUNER_MODE_TV = 0x0001,
AMTUNER_MODE_FM_RADIO = 0x0002,
AMTUNER_MODE_AM_RADIO = 0x0004,
AMTUNER_MODE_DSS = 0x0008 ,
AMTUNER_MODE_DTV = 0x0010

If this method is called with Mode parameter = -2, a Long value with available modes are returned.
If Mode is -1, current tuner mode is returned.

SetTVFormat

Function SetTVFormat (NewFormat As Long) As Long

Set TV format (PAL, NTSC, etc.)

If NewFormat argument is -1, this method returns current TV format.

TV format values are:

AnalogVideo_None = 0x00000000
AnalogVideo_NTSC_M = 0x00000001,
AnalogVideo_NTSC_M_J = 0x00000002,
AnalogVideo_NTSC_433 = 0x00000004,
AnalogVideo_PAL_B = 0x00000010,
AnalogVideo_PAL_D = 0x00000020,
AnalogVideo_PAL_H = 0x00000080,
AnalogVideo_PAL_I = 0x00000100,
AnalogVideo_PAL_M = 0x00000200,
AnalogVideo_PAL_N = 0x00000400,
AnalogVideo_PAL_60 = 0x00000800,
AnalogVideo_SECAM_B = 0x00001000,
AnalogVideo_SECAM_D = 0x00002000,
AnalogVideo_SECAM_G = 0x00004000,
AnalogVideo_SECAM_H = 0x00008000,
AnalogVideo_SECAM_K = 0x00010000,
AnalogVideo_SECAM_K1 = 0x00020000,
AnalogVideo_SECAM_L = 0x00040000,
AnalogVideo_SECAM_L1 = 0x00080000,
AnalogVideo_PAL_N_COMBO = 0x00100000

SetVideoFormat

Function SetVideoFormat (width As Long, height As Long) As Boolean

Set video image dimensions

Returns TRUE if successful or FALSE otherwise.

Because video formats are device-specific, applications should check the return value from this function to determine if the format is accepted by the driver.

SetVideoFormatEx

Function SetVideoFormatEx (CapabilityIndex As Long) As Long

Sets video format by capability index returned by GetVideoCaps method

SetVideoProcAmp

Function SetVideoProcAmp (ValueIndex As Long, nNewValue As Long) As Long

Set video properties

SetZoom

Function SetZoom (left As Long, top As Long, width As Long, height As Long) As Long

Set zoom rectangle on video. Use all zeros as parameters to this method to reset zoom.

```
Dim w, h
w = vcx.VideoWidth / 2
h = vcx.VideoHeight / 2
vcx.SetZoom w / 2, h / w, w, h
```

ShowAudioCodecDlg

Function ShowAudioCodecDlg As Long

Shows audio codec dialog.

Displays a settings dialog provided by video/audio device driver. Must be connected to use this method.

ShowAudioFormatDlg

Function ShowAudioFormatDlg As Long

Shows audio format dialog.

Displays a settings dialog provided by video/audio device driver. Must be connected to use this method.

ShowAudioSourceDlg

Function ShowAudioSourceDlg As Long

Shows audio source dialog.

Displays a settings dialog provided by video/audio device driver. Must be connected to use this method.

ShowTunerDlg

Function ShowTunerDlg As Long

Display TV tuner properties dialog

ShowUserFilterDlg

Function ShowUserFilterDlg (FilterIndex As Long) As Long

Display user filter property page

ShowVideoCodecDlg

Function ShowVideoCodecDlg As Long

Shows video codec dialog.

Displays a settings dialog provided by video/audio device driver. Must be connected to use this method.

ShowVideoCrossbarDlg

Function ShowVideoCrossbarDlg As Long

Display video crossbar dialog

ShowVideoFormatDlg

Function ShowVideoFormatDlg As Long

Shows video format dialog.

Displays a settings dialog provided by video/audio device driver. Must be connected to use this method.

ShowVideoSourceDlg

Function ShowVideoSourceDlg As Long

Shows video source dialog.

Displays a settings dialog provided by video/audio device driver. Must be connected to use this method.

SingleFrameAdd

Function SingleFrameAdd As Long

Adds current video frame into AVI file opened by SingleFrameOpen method

Preview must be enabled.
Single frame capture and real-time capture can't run in the same time.

SingleFrameAddPicture

Function SingleFrameAddPicture (BitmapHandle As Long) As Long

Adds a Windows bitmap to the AVI file created with SingleFrameAdd method.

If GetHBitmap() method of .NET Bitmap class is used to get bitmap handle, it must be deleted after use to avoid memory leaks.

Example:

```
[System.Runtime.InteropServices.DllImport("gdi32.dll")]
public static extern bool DeleteObject(IntPtr hObject);

private void axVideoCapX1_NewFrame(object sender, EventArgs e)
{
pictureBox1.Image = axVideoCapX1.GrabFrame();
{
```

```
IntPtr hb = ((System.Drawing.Bitmap)(pictureBox1.Image)).GetHbitmap();
axVideoCapX2.SingleFrameAddPicture(hb.ToInt32());
DeleteObject(hb);
}
}
```

SingleFrameClose

Function SingleFrameClose As Long

Closes single-frame AVI capture

SingleFrameOpen

Function SingleFrameOpen (fps As Double) As Long

Creates AVI file for single-frame capturing.

Preview must be enabled.
Single frame capture and real-time capture can't run in the same time.
AVI filename is specified by CapFilename property of VideoCapX control.
Fps parameter sets frames-per-second value in new AVI file. Use SingleFrameOpen(5) for 5 frames-per-second AVI.

StartBroadcast

Function StartBroadcast (port As Long, MaxConnections As Long) As Long

Starts WindowsMedia network broadcast at specified port. Use Windows MediaPlayer's OpenURL command to see video on network.

This method starts WindowsMedia broadcast from local PC.
Paramter:
port specifies TCP/IP port number to use for broadcast.
MaxConnections specifies how many clients can connect.
Before starting broadcast, you should select WM profile (bitrate, etc.) by setting ProfileIndex (ProfileData) property.

When a client (Windows Media Player) connects, ConnectionRequest event is raised. When clients ends connection, ConnectionClosed event is raised.

vcx.Connected=TRUE
vcx.Preview=TRUE
vcx.WMAttributes="my title|author is me|copyright to me|rating is 5|this is description "
vcx.StartBroadcast 8080,5

StartBroadcastPush

Function StartBroadcastPush (URL As String, User As String, password As String) As Long

Start sending broadcast to Windows Media server publishing point

String that contains the URL of the publishing point on the Windows Media server. For example, if the URL is "http://MyServer/MyPublishingPoint", the push sink connects to the publishing point named MyPublishingPoint on the server named MyServer. The default port number is 80. If the server is using a different port, specify the port number in the URL. For example, "http://MyServer:8080/MyPublishingPoint" specifies port number 8080.
If the publishing point specified in pwsURL does not exist, the server creates a new publishing point. The caller must have write and create permissions on the server. The new publishing point has the same configuration as the server's default publishing point.

Use **Username** and **Password** parameters to authorize to Windows Media Server.

vcx.WMAttributes="my title|author is me|copyright to me|rating is 5|this is description "
vcx.StartBroadcastPush "http://myserver:8080/pubpoint","mylogin","mypassword"

StartCapture

Function StartCapture As Boolean

Starts video capture

Returns TRUE if successful or FALSE otherwise.
Captured data is saved into file specified in CapFilename property.

Video can be captured in AVI or WMV files. If CapFilename property has .AVI extension, audio/video codec can be specified using AudioCodecIndex/VideoCodecIndex properties. If .WMV file is being captured, audio/video compression is determined by ProfileIndex or ProfileData properties.
WM stream attributes can be set using WMAttributes property.

StopBroadcast

Function StopBroadcast As Long

Stops WM broadcast

StopCapture

Function StopCapture As Boolean

Stops video capture

StoreAutoTune

Function StoreAutoTune As Long

The StoreAutoTune method saves the fine-tuning information for all channels.

UploadFile

Function UploadFile (server As String, username As String, password As String, path As String, server_filename As String, local_filepath As String, [port As Long = 21]) As Boolean

Upload a file to FTP server

UploadFrame

Function UploadFrame (server As String, username As String, password As String, path As String, filename As String, port As Long, quality As Long) As Boolean

Sends current video frame to FTP server

Quality parameter is a JPEG image quality setting (0-100).
Port is usually 21 for FTP service.

See FTPPassiveMode property.

Vcx1.UploadFrame(“ftp.foo.com”,”john”,”tiger”,”images”,”mypic.jpg”,21,70)

VCRSetMode

Function VCRSetMode (Mode As Long) As Long

Control Digital Video VCR. Use this method to change VCR modes of DV camcorder.

Return value of -1 indicates an error or no VCR present.
Return value of 0 indicates no media present in VCR.

Modes are:

1 play
2 stop
3 pause
4 resume
5 fast-forward
6 rewind
7 record
8 record single frame
9 record pause
10 frame-step forward
11 frame-step back
12 play-fastest-forward
13 play-fastest-rev
14 play-slowest-forward
15 play-slowest-rev

To get current VCR mode:
CurMode = VideoCapX.VCRSetMode(0)

To start playing:
VideoCapX.VCRSetMode(1)

Events

CaptureEnd

Sub CaptureEnd

Triggered when capture is ended

CaptureReady

Sub CaptureReady

Raised after StartCapture is called but before any video is actually captured into file. It gives application the opportunity to display 'press to start capture...' message.

CaptureStart

Sub CaptureStart

Triggered when capture is started

DeviceLost

Sub DeviceLost

Raised when device lost is detected. Such as camera removal or cable plug-out.

FullscreenLost

Sub FullscreenLost

Raised when full-screen mode ends due to user action

NewFrame

Sub NewFrame

Raised when new video frame is available

RecompressCompleted

Sub RecompressCompleted

Recompress method runs in background. This event is fired when recompression is finished.

RecompressProgress

Sub RecompressProgress (nPercent As Long, Cancel As Long)

Reports progress of Recompress method processing.