

---

## **VA Control**

---

**Version 1.8.0.6**

2004/10/12

---

---



---

# TABLE of CONTENTS

<b>CHAPTER 1 OVERVIEW .....</b>	<b>7</b>
1.1 INTRODUCTION .....	8
<i>Getting Started with VA Control .....</i>	<i>8</i>
<i>Installing the ActiveX control.....</i>	<i>8</i>
<i>Using the VA control in web pages .....</i>	<i>8</i>
<i>File Structure .....</i>	<i>9</i>
1.2 RELEASE NOTE.....	10
<i>Version 1.8.0.6 .....</i>	<i>10</i>
<i>Version 1.7.0.9 .....</i>	<i>12</i>
<i>Version 1.6.0.1 .....</i>	<i>13</i>
<i>Version 1.4.0.1 .....</i>	<i>14</i>
<i>Version 1.3.0.1 .....</i>	<i>16</i>
<i>Version 1.3.0.0 .....</i>	<i>16</i>
<i>Version 1.1.0.11.....</i>	<i>17</i>
<i>Version 1.1.0.0 .....</i>	<i>17</i>
<i>Version 1.0.0.x.....</i>	<i>18</i>
<b>CHAPTER 2 PROGRAMMER'S GUIDE.....</b>	<b>19</b>
2.1 USING VADECODER WITH YOUR PROJECT .....	20
<i>Adding ActiveX Control to your Project .....</i>	<i>20</i>
<i>Create a Member Variable for VaCtrl.....</i>	<i>20</i>
<i>Set Visual Server contact information to Get Live Video/Audio streams .....</i>	<i>20</i>
<i>Control PTZ camera attached to server.....</i>	<i>21</i>
<i>Save current image to a file .....</i>	<i>21</i>
<i>Get current image data in your program.....</i>	<i>22</i>
2.2 ENUMERATION .....	23
<i>EAudioCodecType.....</i>	<i>24</i>
<i>EClickEventHandler .....</i>	<i>25</i>
<i>EConnectionType.....</i>	<i>26</i>
<i>EConnProtocol .....</i>	<i>27</i>
<i>EControlType .....</i>	<i>28</i>
<i>ECoordinateType .....</i>	<i>29</i>
<i>EDBMediaType.....</i>	<i>30</i>
<i>EDBRecordEventType.....</i>	<i>31</i>

---



---

---

<i>EDBStatusCode</i> .....	32
<i>EDisplayTimeFormat</i> .....	33
<i>EMediaType</i> .....	34
<i>EPictureFormat</i> .....	35
<i>ERegistryRoot</i> .....	36
<i>ESpeedType</i> .....	37
<i>EStreamingOption</i> .....	38
<i>EVideoCodecType</i> .....	39
2.3 APPLICATION SAMPLE CODE .....	40
<b>CHAPTER 3 APPLICATION PROGRAM INTERFACE REFERENCE</b> .....	<b>41</b>
3.1 PROPERTIES .....	42
<i>AudioEnabled</i> .....	42
<i>AudioProtocol</i> .....	43
<i>AutoReconnect</i> .....	44
<i>CircularMode</i> .....	45
<i>ClickEventHandler</i> .....	46
<i>ControlID</i> .....	47
<i>ControlPort</i> .....	48
<i>ConnectionProtocol</i> .....	49
<i>ConnectionTimeout</i> .....	50
<i>ControlType</i> .....	51
<i>CurrentAudioCodecType</i> .....	52
<i>CurrentAudioPort</i> .....	53
<i>CurrentAudioProtocol</i> .....	54
<i>CurrentControlCam</i> .....	55
<i>CurrentMediaType</i> .....	56
<i>CurrentProtocol</i> .....	57
<i>CurrentVideoCodecType</i> .....	58
<i>CurrentVideoPort</i> .....	59
<i>CurrentVideoProtocol</i> .....	60
<i>DatabasePath</i> .....	61
<i>Deblocking</i> .....	62
<i>DecodeAV</i> .....	63
<i>DigitalInURL</i> .....	64
<i>DigitalOutURL</i> .....	65
<i>DigitalZoomEdit</i> .....	66
<i>DigitalZoomEnabled</i> .....	67

---

---

<i>DigitalZoomEnableChk</i> .....	68
<i>DigitalZoomFactor</i> .....	69
<i>DigitalZoomX</i> .....	70
<i>DigitalZoomY</i> .....	71
<i>Display</i> .....	72
<i>DisplayTimeFormat</i> .....	73
<i>DisplayErrorMsg</i> .....	74
<i>DDrawOnePass</i> .....	75
<i>EnableMD</i> .....	76
<i>EventTypes</i> .....	77
<i>ForceGDI</i> .....	78
<i>GDIUseStretchBlit</i> .....	79
<i>GetMD</i> .....	80
<i>GetMDParmUrl</i> .....	81
<i>HttpPort</i> .....	82
<i>HWnd</i> .....	83
<i>IgnoreBorder</i> .....	84
<i>IgnoreCaption</i> .....	85
<i>IndexSize</i> .....	86
<i>JpegQuality</i> .....	87
<i>JpegSecsPerFrame</i> .....	88
<i>JpegURL</i> .....	89
<i>Language</i> .....	90
<i>LeftTitleSpace</i> .....	91
<i>Location</i> .....	92
<i>MaxFileSize</i> .....	93
<i>MaxLocationSize</i> .....	94
<i>MDEditMode</i> .....	95
<i>MediaType</i> .....	96
<i>MediaRecord</i> .....	97
<i>NotifyImageFormat</i> .....	98
<i>NotifyNewAudio</i> .....	99
<i>NotifyVideoData</i> .....	100
<i>Password</i> .....	101
<i>PlayMute</i> .....	102
<i>PlayVolume</i> .....	103
<i>PostEventTime</i> .....	104
<i>PreEventTime</i> .....	105

---

---

<i>PresetURL</i> .....	106
<i>PtzURL</i> .....	107
<i>RecallURL</i> .....	108
<i>ReadWriteTimeout</i> .....	109
<i>ReconnectionWait</i> .....	110
<i>RegkeyRoot</i> .....	111
<i>RegSubKey</i> .....	112
<i>RemoteID</i> .....	113
<i>RemoteIPAddr</i> .....	114
<i>RemotePort</i> .....	115
<i>RightTitleSpace</i> .....	116
<i>SetMD</i> .....	117
<i>SetMDParmUrl</i> .....	118
<i>Stretch</i> .....	119
<i>StreamingOption</i> .....	120
<i>TitleBarColor</i> .....	121
<i>TitleTextColor</i> .....	122
<i>UartURL</i> .....	123
<i>Url</i> .....	124
<i>UserDataFormat</i> .....	125
<i>UserName</i> .....	126
<i>VideoProtocol</i> .....	127
<i>VSize</i> .....	128
3.2 METHODS .....	129
<i>Connect</i> .....	129
<i>Disconnect</i> .....	130
<i>GetConnectionStatus</i> .....	131
<i>GetDigitalIn</i> .....	132
<i>GetPtzPresetPosition</i> .....	134
<i>GetSnapshot</i> .....	136
<i>GetSystemStatus (N/A)</i> .....	137
<i>GetUartData</i> .....	138
<i>GetUartDataBinary</i> .....	139
<i>HttpCommand</i> .....	140
<i>RecallPtzPosition</i> .....	141
<i>RepairDatabase</i> .....	142
<i>RepairLocation</i> .....	144
<i>RestoreControlHandle</i> .....	146

---

---

<i>SavePresetPosition</i> .....	147
<i>SaveSendMail</i> .....	148
<i>SaveSnapshot</i> .....	149
<i>SendCameraControl</i> .....	150
<i>SendCameraControlMap</i> .....	151
<i>SendCameraControlSpeed</i> .....	152
<i>SendDigitalOut</i> .....	153
<i>SendUartCommand</i> .....	154
<i>SendUartCommandBinary</i> .....	155
<i>SetDatabasePath</i> .....	156
<i>SetLocation</i> .....	157
<i>StartMediaRecord</i> .....	158
<i>StartMediaRecordEx</i> .....	159
<i>StopMediaRecord</i> .....	161
3.3 EVENTS .....	162
<i>OnClick</i> .....	163
<i>OnConnectionBroken</i> .....	164
<i>OnConnectionOK</i> .....	165
<i>OnDIDOAlert</i> .....	166
<i>OnMDAlert</i> .....	167
<i>OnNewAudioPiece</i> .....	168
<i>OnNewImage</i> .....	169
<i>OnNewVideo</i> .....	170
<i>OnRecordStatus</i> .....	171
3.4 ERROR CODE LIST .....	172
<b>CHAPTER 4</b> .....	<b>175</b>
4.1 CODE SIZE .....	176

---

---

# Chapter 1

## Overview



# 1.1 Introduction

This document describes the properties and methods supported by the VA ActiveX control. VA ActiveX control, in this release, supports VS/IP 3000 series model products.

## Getting Started with VA Control

The main function of VA Control is to enable viewing directly in Microsoft Internet Explorer. It also enables easy software development using some development tools such as Microsoft Visual Basic, Microsoft Visual C++ and Microsoft C#.

## Installing the ActiveX control

To install the VaDecoder on your PC, just follow these steps:

1. Download the installation file VaDecoder.zip and unpack it. If you get the distribution disc, just find the VaDecoder.exe under lib subdirectory.
2. Run the VaDecoder.exe.
3. The installed directory contains two subdirectories. The control itself is under lib and the cab-file and the sample html file using this cab-file is under cab subdirectory.

## Using the VA control in web pages

You can get video/audio stream from visual server by using the control in your web pages. To access the VA control services, use HTML <OBJECT> tag and specify CLSID for the control. In addition, use the <PARAM> tag to assign property Url location that you want to connect.

The following example illustrates how to get video from 192.168.0.100 in your web page.

---



```
<OBJECT ID="VaDecoder" WIDTH=362 HEIGHT=270
  CLASSID="CLSID: 98264495-6376-443C-9340-2996038BD143"
  CODEBASE="Vactrl.cab#version=1,0,0,0">
<PARAM NAME="Url" VALUE=" http://demo:demo@192.168.0.100/cgi-bin/video.vam ">
</OBJECT>
```

**Note:** Please refer to the sample.html under the cab subdirectory of installed directory for more detail. Replace the followings with your settings:

- “server.domain.hostname” with the IP or domain name + hostname
- “plugin” for CODEBASE to the actual path on your web server.
- “username” and “password” with the user you open for the internet user, usually, this is the demo user in video server.

The other sample miscsample.html allows you to control more on the control by setting properties and calling functions of this control.

## File Structure

FILE	DESCRIPTON
doc\ VA Control.doc	This manual
lib\VaDecoder.exe	The installation file for the control itself and the cab file that signs the control.
sample\CSharpTest	Live video sample code for C#
sample\VbTest	Live video sample code for VB
sample\VsTest	Live video sample code for C++

## 1.2 Release note

### Version 1.8.0.6

- System Requirements

Software:

1. Windows 98/ME/2000/XP

- Added/Enhanced Feature

- ◆ Support multiple monitors now. The control could be moved anywhere without losing of performance.
  - ◆ Controls in the same process will share the same drawing and synchronization thread to gain better performance. But there is maximum limitation for controls now. The upper bound is 128.
  - ◆ Add German/Czech/Korean string.
  - ◆ The control is Unicode-enabled if the running platform is Windows 2K/XP. So the control interface would not show strange symbol if the needed font for the language is installed.
  - ◆ Add [CurrentProtocol](#) to replace [CurrentAudioProtocol](#) and [CurrentVideoProtocol](#).
  - ◆ Add [ConnectionProtocol](#) to replace [AudioProtocol](#) and [VideoProtocol](#).
  - ◆ Add [CurrentMediaType](#) and [MediaType](#) to replace [AudioEnabled](#).
  - ◆ Add [ControlID](#) to let users save the settings for different ActiveX controls in the same program.
  - ◆ Add [RegkeyRoot](#) and [RegSubKey](#) to let users save the settings under different registry key other than standard.
  - ◆ Add [ConnectionTimeout](#) and [ReadWriteTimeout](#) to let users control the timeout value for connecting and read/write.
  - ◆ Add [CurrentVideoCodecType](#) to let users know
-

---

what's the current video codec type.

- ◆ Add [JpegQuality](#), [NotifyImageFormat](#), [NotifyVideoData](#), and [OnNewVideo](#) to let users get the decoded video data.
  - ◆ Add [JpegSecsPerFrame](#), [JpegURL](#), and [StreamingOption](#) to let users use single jpeg mode rather than normal streaming mode to get media data from servers.
  - ◆ Add [LeftTitleSpace](#), [RightTitleSpace](#), [TitleBarColor](#), and [TitleTextColor](#) to let users control more on the appearance of the control.
  - ◆ Add [PlayMute](#), [PlayVolume](#) to let users control playing volume by program.
  - ◆ Add UI to let users control volume when playing.
  - ◆ Add [SaveSendMail](#) to let users save the snapshot and invoke the mailer to send the mail with the snapshot as attachment.
  - Fixed Bugs
    - ◆ UDP reordering is changed to adaptive.
    - ◆ Add Load/Save function for `IpersistStream` so from this version on, update of plugin does not necessary to recompile the program.
    - ◆ Fix the bug for click on image when in 4CIF or 4SIF video mode.
    - ◆ `OnNewImage` is using post rather than send message to prevent dead lock when control end.
    - ◆ Fix the error for window name of motion detection contains special characters.
    - ◆ When motion detection window is first shown, all windows name must be set to it's current value. Fixed.
    - ◆ URL length could now be up to 256 characters.
    - ◆ Fix video real time problem. But it needs also to update server firmware.
    - ◆ Fix the bug that image would sometimes decoded error if the image content is complex.
    - ◆
-

## Version 1.7.0.9

- System Requirements

- Software:

- 2. Windows 98/ME/2000/XP

- Added/Enhanced Feature

- ◆ Add [DisplayTimeFormat](#) to let users decode what type of time format to be displayed on caption.
- ◆ Add [UserDateFormat](#) to let users decode what type of date format to be displayed on caption.
- ◆ Add [MaxLocationSize](#) to let users specify the maximum size a location could use in circular recording mode.
- ◆ Support digital zoom. Please refer to [DigitalZoomEdit](#), [DigitalZoomEnabled](#).
- ◆ The no signal flag in [OnNewImage](#) is now workable if connecting to server with new firmware.
- ◆ Support event record. Please refer to [StartMediaRecordEx](#) and [EventTypes](#).
- ◆ Support database and location repair. Please refer to [RepairDatabase](#), [RepairLocation](#).
- ◆ Add [DecodeAV](#) to let users turn off the decoding in case that they use this control as a recording tool.

- Fixed Bugs

- ◆ Click on image for clickeventhandler = 2 did not work, fixed.
  - ◆ Fix the bug that http mode does not work if the IE security patch is installed.
  - ◆ Click on image would block after several times click if the connecting user does not have permission to control PTZ.
  - ◆ Reconnect sometimes does not work, fixed.
  - ◆ When user name and password contains URLencoding format (%20, for example) it would not connect to server. Fixed.
-

- 
- ◆ Event with Variant \* does not work for some COM server (ATL program for example). Fixed.
  - ◆ Force to reset database and location even if the value is the same, this could let user to apply new setting ([MaxLocationSize](#) for example).
  - ◆ [DisplayErrorMsg](#) had a spelling error in last version, fixed.
  - ◆ Fix motion control panel size problem.
  - ◆ TCP/HTTP would lose first packet due to incorrect internal state. Fixed.
  - ◆ When reconnect, sometimes the control crashes, fixed.
  - ◆ Fix the string truncation in large font mode.
  - ◆ Control would have memory leak sometimes. Fixed.
  - ◆ The strings for multilingual are incorrect, fixed.
  - ◆ Fix current access problem for database.
  - ◆ Motion detection window would show white line under caption in some computer, fixed.

## Version 1.6.0.1

- System Requirements

Software:

3. Windows 98/ME/2000/XP

- Added/Enhanced Feature

- ◆ Video and Audio protocol could now be set from IE, but the get function of these two properties always return 0 to prevent users from using it.
  - ◆ Add ReadSettingByParam to let users set protocol rather reading from registry.
  - ◆ Default will use DirectDraw if possible.
  - ◆ Except in YUY2 mode, all the rectangles of motion alert window are drawn with only one line
  - ◆ Add Ignore Border property to let users disable border (includes caption)
  - ◆ Add GDIUseStretch property to let users choose
-

stretch blt when in GDI mode

- ◆ Add ForceGDI property to let users could force to use GDI mode even if directDraw is available users could switch back and forth by changing this property
- ◆ Add OnePass property for directdraw mode.
- ◆ Synchronize the sample code for C#, C++ and VB.
- ◆ Add more samples for html pages.
- Fixed Bugs
  - ◆ The thread ending waiting is now more sophisticated. Wait for thread to end before going further.
  - ◆ Fix bug that reconnection failed when the users have password set.
  - ◆ GDI without stretching did not work properly before, fixed.
  - ◆ The OnNewImage is not fired in last version, fixed
  - ◆ Fix the bug that the ignore the following addresses in IE is not applied when connecting to server
  - ◆ In GDI mode, the first line is not drawn in last version
  - ◆ The org window name must be updated after successfully save motion setting
  - ◆ Window Size for the motion detection setting panel is now fixed value, so caller could know exactly the width to set.
  - ◆ Fix the bug that motion editing window could be resized when maximized and reload again
  - ◆ Preset position for PT couldn't be retrieved, fixed

## Version 1.4.0.1

- System Requirements
    - Software:
      - 4. Windows 98/ME/2000/XP
  - Added/Enhanced Feature
-

- 
- ◆ Add a property to allow user turn on/off error message box.
  - ◆ Enhance the audio quality by tuning the queue size and the way to drop packets.
  - ◆ Much more real time for video.
  - ◆ Message box now would timeout and close automatically.
  - ◆ Adjust the ID number for property to let the control usable in HTML.
  - ◆ New function to let user send server a command.
  - Fixed Bugs
    - ◆ Motion detection window handle resizing and moving had problem in last version, so the control could crash after saving (due to motion detection window larger than window size), now fixed.
    - ◆ When reconnect, the control will retrieve the motion detection information if in MD mode.
    - ◆ The connection time would be negative if connect failed for the first time. Fixed.
    - ◆ Reconnection will crash in last version, fixed.
    - ◆ Null username & password will be crash when connecting.
    - ◆ Daylight saving handling gave users a “-1” hour when in boundary condition. Fixed.
    - ◆ Server time conversion is changed due to the VNDP definition.
    - ◆ If the connection protocol is changed from TCP to HTTP, the packet reconstruction mechanism is incorrect. Fixed.
    - ◆ Fix the bug that snapshot does not work.
    - ◆ Fix bug that that vbscript or javascript could not receive event.
    - ◆ When using in IE control in VC++ program, click save or new button for MD edit mode window will hang the program. Fixed.
    - ◆ CMOS half size video did not work in last version.
-

Fixed.

- ◆ Half size height is not correct.
- ◆ Make it work to be used in server's main page.
- ◆ X, Y for "click-on-image" should be subtracted by the width of border and caption.
- ◆ Did not turn off "new" button where there are already three MD editing windows.
- ◆ In UDP mode with high bit rate, the video would be broken easily, fixed.

## **Version 1.3.0.1**

- System Requirements

Software:

5. Windows 98/ME/2000/XP

- Fixed Bugs

- ◆ UDP connection will not hang the program now.

## **Version 1.3.0.0**

- System Requirements

Software:

6. Windows 98/ME/2000/XP

- Fixed Bugs

- ◆ Transparent window title font should be changed according to language.
- ◆ For server that use blank as logo or background would be not able to be connected by this control, fixed.

- Changes

- ◆ OnMDAlert and OnNewAudioPiece event parameters are changed to VARIANT rather than VARIANT \*
- ◆ Add one property to decide if OnNewAudioPiece should be callback or not

- Notes

- ◆ Some properties or methods in this version are not working due to the counterpart of video
-



---

server is not implemented yet. We will put a special mark: **N/A** in such properties/methods.

## Version 1.1.0.11

- System Requirements

Software:

7. Windows 98/ME/2000/XP

- Fixed Bugs

- ◆ Url path for video stream is not hard-coded now
- ◆ Under motion detection mode, delete second window crash program.
- ◆ Path of the URL over 40 characters didn't work
- ◆ Return value (plRet) for Connect is not initial to 0 if it already connects to server.
- ◆ Audio quality is bad.
- ◆ It's now able to accept host name + domain name.
- ◆ Program is stuck when load heavy. Now fixed.
- ◆ Proxy setting is working, and a message will show that the protocol change if UDP/TCP is not applicable.
- ◆ SIF size image shown incorrectly.
- ◆ Blocking HTTP operation such as GetPtzPosition is now working.

- Changes

- ◆ GDI support
- ◆ Default protocol is changed to TCP

- Notes

- ◆ Some properties or methods in this version are not working due to the counterpart of video server is not implemented yet. We will put a special mark: **N/A** in such properties/methods.

## Version 1.1.0.0

Add media recording capability

---

## Version 1.0.0.x

- System Requirements

- Software:

- 8. Windows 98/ME/2000/XP

- Features

- ◆ Several handling selections for mouse click event.  
But now only firing event is implemented.
  - ◆ Support reconnection when connection broken.
  - ◆ Controlling for display of received video data.
  - ◆ Controlling for deblocking of decoded image.
  - ◆ Support several camera control functions.
  - ◆ Support reading data from /write data to UART on VS server.
  - ◆ Support events.

- Notes

- ◆ Some properties or methods in this version are not working due to the counterpart of video server is not implemented yet. We will put a special mark: *N/A* in such properties/methods.
-

---

## Chapter 2

# PROGRAMMER'S GUIDE



## 2.1 Using VaDecoder with your project

### Adding ActiveX Control to your Project

To add a control into project's toolbox

1. From the **Project** menu, select **Add To Project/Components and Controls**. The Component and Controls Gallery dialog shows up.
2. Open the **Registered ActiveX Controls** folder. Choose **VaCtrl class** in the list box.
3. Click **Insert** button to close the dialog, and the VaCtrl object will now appear in the toolbox.
4. Now you can drag the VaCtrl object into your dialog.

For VB and C# user, the procedure is quite straightforward. Please refer to the specified user guides for using ActiveX with the programming languages.

### Create a Member Variable for VaCtrl

Right click on the control and choose **ClassWizard**. In the ClassWizard window select the **Member Variables** tab and add a new member variable for the object, for example `m_VaCtrl`.

### Set Visual Server contact information to Get Live

#### Video/Audio streams

The basic operation is to get live images and audio streams from visual server. When using the control, you must set the [RemoteIPAddr](#), [Url](#), [UserName](#) and [Password](#) properties. Optionally, you could also set the HTTP port. Note, you could assemble all the information into [Url](#) property, like the way used in web page, but we recommend you to use the former way. With correct settings above, you could now call Connect

---

---

method to start the download of Video and Audio stream.

Add the following sample code in your **OnInitDialog** function so that the program will start showing live images and playing waves as soon as it starts. The IP address in the [RemotelPAddr](#) must be the correct IP address to a visual server.

```
m_VaCtrl.SetUserName("root");
m_VaCtrl.SetPassword("0002D1000001");
m_VaCtrl.SetRemotelPAddr("192.168.0.100");
m_VaCtrl.SetUrl("/cgi-bin/video.vam");
m_VaCtrl.Connect();
```

## Control PTZ camera attached to server

If you use supported PTZ camera on server, you can control the camera through [SendCameraControl](#) method. Note that you can change the URL for PTZ control by setting [PtzURL](#) property, but usually there is no need to do this. The *option* parameter in the [SendCameraControl](#) function could be "up", "down", "left", "right", "home" for moving the camera, "tele", "wide" for zooming the camera. "far", "near", "auto" for camera focus adjustment (**Note: focus adjustment is not available now**). And the control also has [SendCameraControlMap](#) method that can be used to move the camera to the coordinate you assign (**Note: map control is not available now**). As for the Pan speed and Tilt speed, you could use [SendCameraControlSpeed](#) method to achieve this (**Note: speed control is not available now**). They all use the same URL shown below.

```
// There is a default value, usually, you don't have to set it
m_VaCtrl.SetPtzUrl("http://root:0002D1000400@192.168.0.100/cgi-bin/ camctrl.cgi ");
m_VaCtrl.SnedCameraControl("up");
```

## Save current image to a file

You can save the current image to a local file using Bitmap format. The *pFilename* parameter should be an absolute file path and the other *type* parameter has two option , 2 for bitmap. Other value is not available now.

The following code saves an image to D:\image.jpg file in BMP format.

```
m_VaCtrl.SaveCurrentImage("D:\image.jpg", 2);
```

---

## Get current image data in your program

You can also handle the image data in the program by calling [GetSnapshot](#).

The following code illustrates how to get a bitmap from the control. Note that you have to connect to Visual Server before calling [GetSnapshot](#).

```
VARIANT vData;  
m_VaCtrl.GetSnapshot(2, &vData);
```

---

## 2.2 Enumeration

The enumerations in this section are only available for VB and C#, if you need to pass value of the following enumerations as parameters in VC, please use the corresponding value.

---

---

## EAudioCodecType

### List Member

Name	Value	Description
eAuCodec8K	1	Sample rate is 8K bps.
eAuCodec24K	2	Sample rate is 24K bps.

### Description

This enumeration is used when get current audio CODEC type.

---



---

## EClickEventHandler

### List Member

Name	Value	Description
clickNone	0	Doesn't not handle when user click on control.
clickHandleSelf	1	The control will move the camera to be centered on when user click. This value is working only for new firmware that support click on image.
clickSendEvent	2	The control will fire an event with coordinate to notify that user click on the control.
clickHandleSendEvent	3	This value is the combination of clickHandleSelf and clickSendEvent. This value is working only for new firmware that support click on image.

### Description

This enumeration is used with the [ClickEventHandler](#) property.

---

---

## EConnectionType

### List Member

Name	Value	Description
eConnVideo	1	This is the video connection.
eConnAudio	2	This is the audio connection.

### Description

This enumeration is used when connection status (OK/Broken) event fired.

---

---

## EConnProtocol

### List Member

Name	Value	Description
eProtNone	0	The control is currently not connecting to visual server.
eProtUDP	1	Use UDP as the connecting protocol.
eProtTCP	2	Use TCP as the connecting protocol.
eProtHTTP	3	Use HTTP as the connecting protocol. HTTP protocol does not support audio.

### Description

This enumeration is used when set/get audio or video protocol.

---

---

## EControlType

### List Member

Name	Value	Description
eCtrlNoCtrlBar	0	This setting asks the control not to display control bar.
eCtrlNormal	1	The control will show control bar and the image together
eCtrlMotion	2	The control will be shown in motion detection edit-mode. In such mode, no control bar will be shown.
eCtrlClientSet3K	100	The control will appear as a client setting to let users select the protocol for connection.

### Description

This enumeration is used to specify the control UI type. Note: if the control type is switched dynamically, the setting could not be updated if the connection is not re-established.

---

---

## ECoordinateType

### List Member

Name	Value	Description
eCoordAbsolute	1	The coordinate value is related to the upper-left corner of the map.
eCoordRelative	2	The coordinate value is related to current camera position.

### Description

This enumeration is used when control the camera by map coordinates.

---

---

## EDBMediaType

### List Member

Name	Value	Description
eMediaAudio	1	Audio only media type.
eMediaVideo	2	Video only media type.
eMediaAudioVideo	3	Both audio and video present in the media

### Description

This enumeration is used when [StartMediaRecord](#) and [StartMediaRecordEx](#) method.

---

---

## EDBRecordEventType

### List Member

Name	Value	Description
eMDAlertWin1	1	Motion is detected for the first motion detection window.
eMDAlertWin2	2	Motion is detected for the second motion detection window.
eMDAlertWin3	4	Motion is detected for the first third detection window.
eDILow1	256	Digital input is low (the recording will keep until DI is not low)
eDIHigh1	65536	Digital input is high (the recording will keep until DI is not high)
eDIRise1	2097152	Digital input is changed from low to high
eDIFall1	33554432	Digital input is changed from high to low

### Description

This enumeration is used in [EventTypes](#) property.

---

---

## EDBStatusCode

### List Member

Name	Value	Description
eStatusDiskFull	1	Disk is full when recording. The recording will be stopped when this status is called back.
eStatusDBRepairFinish	2	The database repair is finished. The IParam of the event means whether the repair success or not. Nonzero means success, 0 means not.
EStatusLocRepairFinish	3	The location repair is finished. The IParam of the event means whether the repair success or not. Nonzero means success, 0 means not.

### Description

This enumeration is used in [OnRecordStatus](#) event.

---



---

## EDisplayTimeFormat

### List Member

Name	Value	Description
eTimeFmtNormal	0	The normal 24 hours format
eTimeFmtTwelves	1	12 or 24 hours format decided by the system setting in regional control panel. For 12 hours, the time marker is always “AM”/”PM” no matter what language the OS is.
eTimeFmtUser	2	12 or 24 hours format decided by the system setting in regional control panel. For 12 hours, the time marker is the same as the system setting.

### Description

This enumeration is used in [DisplayTimeFormat](#) property.

---

---

## EMediaType

### List Member

Name	Value	Description
eMediaNone	0	There is no media now.
eMediaVideo	1	Only video data is meaningful for the control.
eMediaAudio	2	Only audio data is meaningful for the control.
eMediaAV	3	Both video and audio are meaningful for the control.

### Description

This enumeration is used to specify the media to be got from servers.

---

---

## EPictureFormat

### List Member

Name	Value	Description
ePicFmtJpeg	1	JPEG format.
ePicFmtBmp	2	Bitmap format. The first scanline is on bottom.
ePicFmtYUV	3	YUY2 format. It's ordering is Y1U1Y2V2Y3U3Y4U4...
EPicFmtRaw24	4	RGB24 format. The first scanline is on top.

### Description

This enumeration is used when get image from control or for video data notification. For snapshot, only bmp is available.

---

## ERegistryRoot

### List Member

---

Name	Value	Description
eRegLocalMachine	0	The registry key root is set to local machine.
eRegCurrentUser	1	The registry key root is set to current user.

### Description

---

This enumeration is used in [RegkeyRoot](#) property.

---

---

## ESpeedType

### List Member

Name	Value	Description
ePanSpeed	1	This is the pan speed for camera control
eTiltSpeed	2	This is the tilt speed for camera control.

### Description

This enumeration is used when control the camera speed.

---

---

## EStreamingOption

### List Member

Name	Value	Description
eStOpStreaming	1	Use streaming to get the media.
eStOpSingleJpeg	2	Get single jpeg from server by CGI. It would be slower and all the extra information such as DI/DO and motion information that are carried in stream would not be available in such case.

### Description

This enumeration is used to specify the streaming option when playing live media.

---

---

## EVideoCodecType

### List Member

Name	Value	Description
eViCodecMJpeg	1	Video codec is motion jpeg
eViCodecMpeg4	2	Video codec is MPEG4 simple profile
eViCodecSHM	3	Video codec is MPEG4 short header mode

### Description

This enumeration is used when get current video CODEC type.

---

## 2.3 Application Sample Code

Please refer to the distribution package for the sample code.

---



---

# Chapter 3

# 3

## Application Program

## Interface Reference

This chapter contains the API function calls for the VA Control.

---

## 3.1 Properties

### AudioEnabled

A property to control whether should the control download the audio stream from server.

#### Type

*Boolean*

True means to download the audio, False means not. Default value is true.

#### Attribute

*R/W*

#### Remarks

Once change, the control will reconnect to the server dependent on the new value. When audio is disabled, the control will use HTTP to connect to server. When it is enabled, the control will use the video/audio protocol set now to decide which protocol to use. If the protocol is set to HTTP, audio would be disabled.

---

---

## AudioProtocol

The audio protocol used to communicate with Visual Server.

### Type

#### *Enumeration*

The enumeration now contains three elements: HTTP, UDP, and TCP. Default value is UDP.

### Attribute

W/O

### Remarks

Once set, the control will restart the audio connection if it is currently connecting to server. But there is no guarantee that the connection will use the protocol assigned, it could switch to other protocol if necessary. User could get the [CurrentAudioProtocol](#) to know the protocol actually used.

Note that this property is currently equivalent to [VideoProtocol](#) since Visual Server now only allows video/audio to use the same protocol.

---

## AutoReconnect

Decide whether the control should try to reconnect to the server if it found the connection broken.

### Type

*Boolean*

True means to reconnect automatically, False means not.

### Attribute

*R/W*

### Remarks

No matter what value this property is set, the control will send out a connection-broken event. The event receiver should not try to do reconnection if this value is set to TRUE since the control itself will handle the reconnection.

---

---

## CircularMode

This property decides if the control should delete the oldest file in current location if the file system full.

### Type

*Boolean*

True means to be circular, False means not.

### Attribute

*R/W*

### Remarks

This property should be used very careful, or it could cause lost of data.

---

## ClickEventHandler

Tell the control how to response to the mouse click event.

### Type

*EClickEventHandler*

HandleSelf is the default setting.

### Attribute

*R/W*

### Remarks

Please refer to [EClickEventHandler](#) for the possible value of this property.

---

---

## ControlID

This is the control ID used to write or read registry.

### Type

*Long*

This is the control ID

### Attribute

*R/W*

### Remarks

If the control ID is 0, the registry is read from or written to the default registry root.  
If the value is not 0, the registry is read from or written to a sub-key under the registry root. The sub-key name is constructed base on the control ID.

---

## ControlPort

Tell the control the control port to use to communicate with the Visual Server.

Note: This is a hidden property that should only be used when the control is embedded in IE.

### Type

*Long*

This is the port number.

### Attribute

*R/W*

---



---

## ConnectionProtocol

Set the connection protocol used when connecting to server.

### Type

*EConnProtocol*

The available protocol is listed in [EConnProtocol](#).

### Attribute

W/O

### Remarks

Once set, the control will restart the connection if it is currently connecting to server. But there is no guarantee that the connection will use the protocol assigned, it could switch to other protocol if necessary. User could get the [CurrentProtocol](#) to know the protocol actually used.

Note that this property is currently equivalent to [VideoProtocol](#) and [AudioProtocol](#) since Video Server or IP camera now only allow video/audio to use the same protocol.

---

## ConnectionTimeout

Set or get the timeout value used to connection to server.

### Type

*Long*

The value is in milliseconds.

### Attribute

*R/W*

### Remarks

---

---

## ControlType

This property is used to change the different UI supported by the control. This property is better not to be changed after control created. That is, if the control is used in IE, the value should be set in parameter list, and if used in non-script language, the property should be set in designed time.

### Type

*EControlType*

This is the type for the control. If it is changed at runtime, the control will switch to the new interface.

### Attribute

*R/W*

---

## CurrentAudioCodecType

Read from local the codec type uses by the server.

### Type

*EAudioCodecType*

The elements: Audio8K and Audio24K.

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server, and the [AudioEnabled](#) property should also be set to true.

---

---

## CurrentAudioPort

Read from local the audio port uses by server to provide audio stream service.

### Type

*Long*

This is the port number.

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server, and the [AudioEnabled](#) property should also be set to true.

---

## CurrentAudioProtocol

Read from local the audio protocol uses to connect to server.

### Type

*EConnProtocol*

The enumeration now contains four elements: None, HTTP, UDP, and TCP.

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server, and the [AudioEnabled](#) property should also be set to true.

---

---

## CurrentControlCam

Read or set the current control camera index. This property is reserved for future used. Currently only ID 1 is applicable.

### Type

*long*

the camera index.

### Attribute

*R/W*

### Remarks

---

## CurrentMediaType

Read from local the current media type that is available for the connection to server.

### Type

*EMediaType*

This is media type. The value is one of the value in [EMediaType](#).

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server.

---



---

## CurrentProtocol

Read from local the protocol uses to connect to server. This is the same as [CurrentAudioProtocol](#) and [CurrentVideoProtocol](#) are.

### Type

*EConnProtocol*

The enumeration now contains four elements: None, HTTP, UDP, and TCP.

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server.

---

## CurrentVideoCodecType

Read from local the video codec type uses by the server.

### Type

*EVideoCodecType*

The elements are listed in [EVideoCodecType](#).

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server.

---

---

## CurrentVideoPort

Read from local the video port uses by server to provide video stream service.

### Type

*Long*

This is the port number.

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server.

---

## CurrentVideoProtocol

Read from local the video protocol uses to connect to server.

### Type

*EConnProtocol*

The enumeration now contains three elements: HTTP, UDP, and TCP.

### Attribute

*R/O*

### Remarks

This value is meaningful only after connecting to the server.

---

## DatabasePath

Get the database path that is currently set to the control.

### Type

*String*

This is a string that represents the database path. The path format is the same as what you use under windows explorer.

### Attribute

*R/O*

---

## Deblocking

Get or set the deblocking mode when decode video stream.

### Type

*Boolean*

True means to de-block the video image, False means not. Default value is False.

### Attribute

*R/W*

### Remarks

---

---

## DecodeAV

Get or set the flag that decides if the audio and video data should be decoded after received.

### Type

*Boolean*

True means to decode the audio and video data. False means not decode.

### Attribute

*R/W*

### Remarks

The default value is to decode audio and video data. If the value is changed to false, no audio would be heard and no video data would be shown on screen. But the screen would still show the server time and location name if the Display flag is not turned off.

---

## DigitalInURL

Set or get the URL for digital input.

### Type

*String*

This URL could include or exclude the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---



## DigitalOutURL

Set or get the URL for digital input/output.

### Type

*String*

This URL could include or exclude the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

## DigitalZoomEdit

Set or get the digital zoom edit property. This property controls whether the control panel for digital zoom setting is viewable or not. Do no confuse this property with [DigitalZoomEnabled](#) property.

### Type

*Boolean*

True means the digital zoom control panel is viewable. False means it's hidden.

### Attribute

*R/W*

### Remarks

---

---

## DigitalZoomEnabled

Set or get the digital zoom enabled property. This property controls if the digital zoom function is enabled when displaying video. Do not confuse this property with [DigitalZoomEdit](#) property.

### Type

*Boolean*

True means the digital zoom function is enabled. False means it's disabled.

### Attribute

*R/W*

### Remarks

It is possible to enable/disable digital zoom function even if the edit interface is not shown.

---

## DigitalZoomEnableChk

Set or get the property that controls if the “enable digital zoom” check box is shown or hidden in the zoom control panel.

### Type

*Boolean*

True means check box is shown. False means it's hidden.

### Attribute

*R/W*

### Remarks

When the check box is hidden, the digital zoom enabling or disabling could only be set through the property [DigitalZoomEnabled](#). Users will not able to disable or enable digital zoom on UI.

---

---

## DigitalZoomFactor

Set or get the digital zoom factor property. This property controls the proportion of the video frame to be zoomed.

### Type

*Long*

The range is between 100 and 400. Value outside this range would be normalized to 100 when setting.

### Attribute

*R/W*

### Remarks

The size of the viewable window is  $(\text{Real Video Size} * 100 / \text{this factor})$ . And the position of the viewable window might change when set a new factor. The rule is to enlarge or shrink the window centered at the original rectangle center. If any side of the viewable window is outside the real video, the window is moved to align the real video frame at that side.

---

## DigitalZoomX

Set or get the x coordinate of the left-upper corner of the viewable window. This and the DigitalZoomY property decide which portion of the video is viewable on screen.

### Type

*Long*

The value range is between 0 and the width of half mode video size. So it depends on which kind of lens the server uses.

### Attribute

*R/W*

### Remarks

The control will adjust the value if the value is out of range.

---

---

## DigitalZoomY

Set or get the y coordinate of the left-upper corner of the viewable window. This and the DigitalZoomX property decide which portion of the video is viewable on screen.

### Type

*Long*

The value range is between 0 and the height of half mode video size. So it depends on which kind of lens the server uses.

### Attribute

*R/W*

### Remarks

The control will adjust the value if the value is out of range.

---

## Display

Set or get the switch to turn on or turn off the display of graph on screen.

### Type

*Boolean*

True means to show the graph, False means to disable display.

### Attribute

*R/W*

### Remarks

For computer that has slower graphic card, this switch would be very helpful to prevent the system loading from being filled up.

---



---

## DisplayTimeFormat

Set or get the format to display server time on control title.

### Type

*EDisplayTimeFormat*

The enumeration elements are list in [EDisplayTimeFormat](#).

### Attribute

*R/W*

### Remarks

For eTimeFmtTwelves format, the position of “AM”/”PM” is always after the hour/minute/second string. For the eTimeFmtUser, the position time marker is the same as what users see in regional control panel.

---

## DisplayErrorMsg

Set or get the switch to turn on or turn off the display of error message when error happens.

### Type

*Boolean*

True means to show the message with message box. False means to disable display of error message. The default value is True.

### Attribute

*R/W*

### Remarks

Sometimes application need not to show users the error message could turn it false. But now the message boxes are all timeout box that would be closed automatically after 20 seconds.

---

---

## DDDrawOnePass

Set or get the switch to decide the way to display video when in DirectDraw mode. The default value is False.

### Type

*Boolean*

True means to show the video directly without shivering proof, False to provide shivering proof.

### Attribute

*R/W*

### Remarks

When drawing with DirectDraw, if the video is stretching, the video will be shivering when other window moving upon. The control will use some mechanism to cancel this shivering, but this mechanism won't work in some display card (the video looks bad if stretching or shrinking). The developing could turn this property one when the video quality is bad.

---

## EnableMD

Set the motion detection editing mode to be enabled or disabled.

### Type

*Boolean*

True means to enable edit mode, False to disable it. Default value is disabled.

### Attribute

*W/O*

### Remarks

This property is hidden. It should only be used for control embedded in IE. Other user should use [MDEditMode](#) instead.

---

---

## EventTypes

Set or get the types that will affect the event recording. The event

### Type

*Boolean*

True means to enable edit mode, False to disable it. Default value is disabled.

### Attribute

*W/O*

### Remarks

This property is hidden. It should only be used for control embedded in IE. Other user should use [MDEditMode](#) instead.

---

## ForceGDI

Should the control show video in GDI mode no matter the card supports DirectDraw or not.

### Type

*Boolean*

Set this value to True to force to display video in GDI mode. Default value is False.

### Attribute

*R/W*

### Remarks

GDI has a worse performance but better compatibility than DirectDraw upon display card. On the machine that has problem using DirectDraw, this property could be turned on.

---

---

## GDIUseStretchBlt

Set or get the property that controls the underlying display function when showing video. When this property is turned on, the video quality will be better, but the performance is worse.

### Type

*Boolean*

True to gain better video quality, False to gain better performance.

### Attribute

*R/W*

### Remarks

This property is only usable when the [ForceGDI](#) is set to True or when the machine does not support DirectDraw. If the control runs in DirectDraw mode, this property is ignored. **Note: when the control size is small so that video is shrinking, the motion detection window would sometimes not seen when this property is not set to True. This is Windows API limitation. Turn on this property would resolve this problem.**

---

## GetMD

Get or set the motion detection parameters retrieving URL.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

This property is hidden It should only be used for control embedded in IE. Other user should use [GetMDParmUrl](#) instead.

---



---

## GetMDParmUrl

Get or set the motion detection parameters retrieving URL.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

## HttpPort

Tell the control the http port to use to communication with the Visual Server.

### Type

*Long*

This is the port number. Default value is 80. If you change the HTTP port used by Visual Server, you should also update this value.

### Attribute

*R/W*

### Remarks

Note this value is used to append to the several URLs properties if they are lack of the IP part.

---

---

## HWnd

Get or set the drawing target window handle.

### Type

*Long (HWND)*

The window handle used to draw.

### Attribute

*R/W*

### Remarks

Note this value is originally set to the handle of the window of the control. Once the value is changed, the original window will be hidden. To restore the original window, please refer to [RestoreControlHandle](#).

---

## IgnoreBorder

Get or set if the control should display the border when showing video. The border is the 5-pixel wide gray line enclosing the video. When the value is True, the border will be ignored. The default value is False.

### Type

*Boolean*

Show or hide the border

### Attribute

*R/W*

### Remarks

When in Motion editing mode, the border could not be ignored

---

---

## IgnoreCaption

Get or set if the control should display the caption when showing video. The border is the 20 pixels high text line above the video (and also above the border if any). When the value is True, the caption will be ignored. The default value is False.

### Type

*Boolean*

Show or hide the caption

### Attribute

*R/W*

### Remarks

When in Motion editing mode, the caption could not be ignored.

---

## IndexSize

Get or set the index number use in a media file. The media file size is controlled by two factors: MaxFileSize and IndexSize. Whenever one condition reaches, the media file will be closed, and a new media file will be opened and used.

### Type

*Long*

The index size value.

### Attribute

*R/W*

### Remarks

---

---

## JpegQuality

Get or set the quality value for the jpeg notified by the control.

### Type

*Long*

The value should be within 1-125. The larger value means worse quality.

### Attribute

*R/W*

### Remarks

If the control uses single jpeg mode as its streaming mode, this property would not affect the quality of the jpeg notified. Because the quality is determined when server generates the file. There is no reason to re-generate a jpeg with worse quality in client side.

---

## JpegSecsPerFrame

Get or set the period to retrieve jpeg from server if the streaming mode is single jpeg.

### Type

*Long*

This is the value in seconds. The default value is 1 second.

### Attribute

*R/W*

### Remarks

Please do not set the value too large, or the control would seem to be frozen.

---



---

## JpegURL

Get or set the URL for single jpeg mode to retrieve file.

### Type

*String*

This is the URL to retrieve jpeg. It could be full path or partial path. For partial path, the [RemoteIPAddr](#) must also be set.

### Attribute

*R/W*

### Remarks

The default value is good enough for use with the video server and IP camera. But if users intend to use with proxy server that could cache jpeg frames, this property could not be changed to satisfy such condition.

---

## Language

Get or set the language used when display message or for caption used for the controls in motion detection.

### Type

*String*

Currently four languages are support: English (EN), Traditional Chinese (TC), Japanese (JP) and Simplified Chinese (SC). Default language is English.

### Attribute

*R/W*

### Remarks

The control will get the language setting for the server automatically. But if the protocol is HTTP, the automation will be failed.

---

---

## LeftTitleSpace

Get or set the space between left border and the “text on video” text when drawing caption of the control. The unit is in pixels.

### Type

*Long*

The default value is 2.

### Attribute

*R/W*

### Remarks

---

## Location

Get current location setting for database. A location is a subfolder that saves the media files for a certain server (We don't recommend you to save several servers' data under a same location).

### Type

*String*

Any characters that are permitted for directory path are legal for location.

### Attribute

*R/O*

### Remarks

The length is dependent on database path length. Usually, you won't give a location longer than 32 characters.

---

---

## MaxFileSize

Get or set the maximum file size of one media file.

### Type

*Long*

### Attribute

*R/O*

### Remarks

The unit is in bytes. So you could give at most 2G bytes for one file. But this is not recommended since once file system error, the whole file may be lost. But for size too small, there will be another problem that the maximum file numbers under one subdirectory may be reached easily.

---

## MaxLocationSize

Get or set the maximum size a location allows to save media data when recording in circular mode. This property is ignored when circular mode is disabled.

### Type

*Long*

### Attribute

*R/O*

### Remarks

The unit is in kilobytes. So you could give at most 1T bytes for one location. This would be sufficient for current file system. The default value is 30M bytes, and the minimum value is 5M bytes.

---

---

## MDEditMode

Get or set the motion detection edit mode.

### Type

*Boolean*

True to enable the edit mode, False to disable the edit mode. The former will show the window and meter, buttons.

### Attribute

*R/W*

### Remarks

When this property is changed to be true, the control will connect to the server to get the last setting of motion detection automatically. When the edit mode is turned off, all unsaved state will be lost.

---

## MediaType

Set the media type for connection. This property is the equivalent to call [AudioEnabled](#).

### Type

*EMediaType*

This is the new media type used by this control. The value should be either eMediaVideo or eMediaAV.

### Attribute

W/O

### Remarks

Once change, the control will reconnect to the server dependent on the new value. When audio is disabled, the control will use HTTP to connect to server. When it is enabled, the control will use the video/audio protocol set now to decide which protocol to use. If the protocol is set to HTTP, audio would be disabled.

---



---

## MediaRecord

Get the recording state of the control

### Type

*Boolean*

True means the recording flag of the control is turned on now. False means the control's recording flag is turned off.

### Attribute

*R/O*

### Remarks

To set the recording flag, please use [StartMediaRecord](#) and [StopMediaRecord](#) methods.

---

## NotifyImageFormat

Set or get the image type to be notified when new video data arrive. This property will be meaningful only if [NotifyVideoData](#) is set to be true.

### Type

*EPictureFormat*

The available format is defined in [EPictureFormat](#).

### Attribute

*R/W*

### Remarks

If the format is set the Jpeg, and the streaming option is set to normal streaming rather than single jpeg mode, the control needs to decode the video and encode it to jpeg. So the performance of the control would be even worse than just notify the decoded data (such as bmp24 or YUV). So please set this property carefully.

---

---

## NotifyNewAudio

Determine if the control should send control owner new audio piece arrival event.

### Type

*Boolean*

True means to notify for each piece. False means not.

### Attribute

*R/W*

### Remarks

Because audio pieces are sent by byte array and it could be as large as 640 bytes per piece. Notify too frequently would cause a performance penalty. The default value is False. Note: this callback could not work for script language such as javascript because it uses byte array rather than variant array to saving the audio pieces. And for javascript, it could not handle byte array.

---

## NotifyVideoData

Determine if the control should send control owner new video decoded data by event.

### Type

*Boolean*

True means to notify for each piece. False means not.

### Attribute

*R/W*

### Remarks

Because video pieces are sent by byte array and it could be more than 10K bytes sometimes. Notify too frequently would cause a performance penalty. The default value is False. Note: this callback could not work for script language such as javascript because it uses byte array rather than variant array to saving the video pieces. And for javascript, it could not handle byte array.

---

## Password

Get or set the password used for web page authentication.

### Type

*String*

### Attribute

W/O

### Remarks

For security reason, this attribute is not retrievable.

---

## PlayMute

Turn on or off the audio when playing.

### Type

*Boolean*

True means to turn on the audio. And 'false' means to turn it off.

### Attribute

*R/W*

### Remarks

This property could be changed from control's audio setting panel. But if the panel is dropped down and the value is changed by program, the value will not be reflected until the panel is closed and reopen again.

---

---

## PlayVolume

Get or set the audio volume used when playing.

### Type

*Long*

The value range is 0-100. 0 is equivalent to mute and 100 is the loudest level.

### Attribute

*R/W*

### Remarks

This property shares the same value set in the audio setting panel of the control bar. So if users change the value from UI, this property will be changed. As [PlayMute](#) is, change of the value will be reflected in next time the panel dropped down. Note: the volume value is for DirectSound play buffer, not global to the system. So it might happen that the sound is small even if the property is in maximum level. In such case, please adjust the global volume level from Windows' control panel.

---

## PostEventTime

Get or set the time in seconds that would keep recording after a event happens.

### Type

*Long*

This is the seconds to keep recording after event.

### Attribute

*R/W*

### Remarks

The default value is 5 seconds. The recording will not stop exactly on this value after event. If there is not other event happens before the recording stops, the recording will stop at  $N + M$  seconds. Where  $N$  is  $\text{PostEventTime} - 1$  and  $M$  depends when the 'I' frame is seen. If the 'I' frame appears right after  $N$ , then the recording stops at  $N$  seconds. The maximum value for  $M$  is 4.

If there are other events happen before recording stop, the recording will not stop until the stop criteria match the time setting for the latest event.

---



---

## PreEventTime

Get or set the time in seconds that the control keeps the data before event happens.

### Type

*Long*

This is the seconds to keep recording before event.

### Attribute

*R/W*

### Remarks

The default value is 5 seconds. The control pre-buffers the data in temporary files. These files are removed if no event triggered after PreEventTime seconds. The kept media length will not be exactly PreEventTime seconds, but usually it will be longer. It's possible to have shorter pre-event recording if the event happens right after the control starts to pre-buffer.

---

## PresetURL

Get or set the URL for save camera position.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

---

## PtzURL

Get or set the URL for camera PTZ control.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

## RecallURL

Get or set the URL for recall camera position.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

---

## ReadWriteTimeout

Get or set the timeout value for read/write from network.

### Type

*Long*

This is the timeout value.

### Attribute

*R/W*

### Remarks

This timeout value is used after the connection is established to server. Please do not be confused with [ConnectionTimeout](#).

---

## ReconnectionWait

Get or set the time to wait before the control tries to reconnect to server.

### Type

*Long*

Time to wait in milliseconds. Value less than or equals to 0 means to reconnect at once. Default value is 30000 milliseconds.

### Attribute

*R/W*

---

---

## RegkeyRoot

Get or set the registry root for saving control settings. It determines whether to save registry under HKEY\_LOCAL\_MACHINE or under HKEY\_CURRENT\_USER.

### Type

*ERegistryRoot*

The default value is eRegLocalMachine, which means to save under HKEY\_LOCAL\_MACHINE.

### Attribute

*R/W*

### Remarks

This property lets users could save the setting under their own registry hierarchy.

---

## RegSubKey

Get or set the registry base for saving control settings. The base starts from the root specify by [RegkeyRoot](#).

### Type

*String*

### Attribute

*R/W*

### Remarks

This property lets users could save the setting under their own registry hierarchy.

---



---

## RemotelD

Get or set the control id used by current session.

### Type

*Long*

### Attribute

*R/W*

### Remarks

Hidden property that is only used under IE. For developing stand along program, there is no need to set this value.

---

## RemoteIPAddr

Get or set the remote IP address of the Visual Server.

### Type

*String*

The format should be dotted IP, that is: 'a.b.c.d'.

### Attribute

*R/W*

### Remarks

If used in Internet Explorer, this value will be retrieved automatically. There is no need to set this value.

---

---

## RemotePort

Get or set the remote control port of the Visual Server.

### Type

*Long*

### Attribute

*R/W*

### Remarks

This property is set to hidden since it only means to be used in Internet Explorer.

---

## RightTitleSpace

Get or set the space between right border and the “server time” text when drawing caption of the control. The unit is in pixels.

### Type

*Long*

The default value is 2.

### Attribute

*R/W*

### Remarks

---

---

## SetMD

Get or set the motion detection parameters setting URL.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

This property is hidden It should only be used for control embedded in IE. Other user should use [SetMDParmUrl](#) instead.

---

## SetMDParmUrl

Get or set the motion detection parameters setting URL.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

---

## Stretch

Tell the control to stretch the graph to the size of the display window or not.

### Type

*Boolean*

True means to stretch, False means not.

### Attribute

*R/W*

### Remarks

If the graph is not stretched, and the view part of the control is greater than the video frame, the frame would be put align to the upper left corner of the control. If the view part is smaller than the video frame, the center point of the view part and the frame would be put together.

---

## StreamingOption

Set or get the streaming option for the control.

### Type

*EStreamingOption*

The available options are listed in [EStreamingOption](#).

### Attribute

*R/W*

### Remarks

When the streaming option is changed, the connection will be re-established.

---



---

## TitleBarColor

Get or set the background color used to draw the caption.

### Type

*OLE\_COLOR*

This is the color value. The default color is black (R:0, G: 0, B:0)

### Attribute

*R/W*

### Remarks

---

## TitleTextColor

Get or set the text color used to draw the caption.

### Type

*OLE\_COLOR*

This is the color value. The default color is white (R:255, G: 255, B:255)

### Attribute

*R/W*

### Remarks

The color is only applicable to control in non-recording mode. When in recording mode, the text color is set to RED(R: 255, G: 0, B: 0). So please do not set [TitleBarColor](#) to RED or the caption would be not visible when recording.

---

---

## UartURL

Set or get the URL for UART control.

### Type

*String*

This URL could be included or excluded the host IP.

### Attribute

*R/W*

### Remarks

The default value works fine for Visual Server. Usually, you don't need to change this value.

---

## Url

Set or get the URL for video retrieving.

### Type

*String*

This URL could include or exclude the host IP. There is no default value, user must give a value, usually, this is “/cgi-bin/video/vam”.

### Attribute

*R/W*

---

---

## UserDateFormat

Set or get the display server date format property.

### Type

*Boolean*

True means to use the format set in regional control panel. False means to use the yyyy/MM/dd format.

### Attribute

*R/W*

### Remarks

The default value is False.

---

## UserName

Set or get the user name for web authentication.

### Type

*String*

### Attribute

*R/W*

### Remarks

This value could also be appended to the [Uri](#) property. The control will parse and retrieve it from the string you given.

---

---

## VideoProtocol

Get or set the video protocol used to communicate with Visual Server.

### Type

*EConnProtocol*

The enumeration now contains three elements: HTTP, UDP, and TCP. Default value is UDP.

### Attribute

W/O

### Remarks

Once set, the control will restart the video connection if it is currently connecting to server. But there is no guarantee that the connection will use the protocol assigned, it could switch to other protocol if necessary. User could get the [CurrentVideoProtocol](#) to know the protocol actually used.

Note that this property is currently equivalent to [AudioProtocol](#) since Visual Server now only allows video/audio to use the same protocol.

---

## VSize

This is the video size of the gotten picture. A Visual Server will send out one size of video at one times no matter how many clients connect to it.

### Type

*String*

Now the value could be "SIF", "QSIF", "QSIF2"

### Attribute

*R/W*

### Remarks

This property is hidden and most of the time it is used only when the control is embedded in the IE.

---



Because the OLE will raise exception when the return value of method is not S\_OK, we always return S\_OK for each method. And we add a parameter that is a pointer of long integer to hold the error code. Now when use these methods, just treat them as functions with return code of type long.

Connect to the Visual Server to start the download.

```
HRESULT Connect ( Long *pRet );
```

Always S\_OK.

*plRet*  
[out] the return code of the function. 0 means success, others mean failed.

If the connection is already established, this call will do nothing and return 0.

The [RemoteIPAddr](#), [HttpPort](#), [Url](#), [VideoProtocol/AudioProtocol](#) value should be ready before calling this method.

## Disconnect

Disconnect to the Visual Server to stop the download.

### Syntax

```
HRESULT Disconnect ( Long *pIRet );
```

### Return Value

Always S\_OK.

### Parameters

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

If the connection is not connected, this call will do nothing and return 0.

---

---

## GetConnectionStatus

Get the connection status.

### Syntax

```
HRESULT GetConnectionStatus ( Variant *pvData,  
                                Long *plRet );
```

### Return Value

Always S\_OK.

### Parameters

*pvData*

[out] The buffer that holds the returned data. It's an array of type Long.

0: Frame Rate

1: Reconnect Times

2: Live Time

3: Total Connection Time

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## GetDigitalIn

Get the digital input value for the specified port.

### Syntax

```
HRESULT GetDigitalIn (    Long IPort,  
                          Long* pData,  
                          Long *pRet                );
```

### Return Value

Always S\_OK.

### Parameters

*IPort*

[in] Specify the port index of the digital input device to get data. Port index starts from 0. This parameter is now reserved for future use.

*pIData*

[out] The buffer that holds the returned data. 0 is for low-level signal. 1 is for high-level signal. Other values are reserved for future use.

*pRet*

[out] the return code of the function. 0 means success, others mean failed.

---

**Remarks**

This function works in blocking mode. It will not return until the data got or HTTP time out occurred.

---

## GetPtzPresetPosition

Retrieve the list of name of preset positions of camera.

### Syntax

```
HRESULT GetPtzPresetPosition ( VARIANT *pvData,  
                                Long *plRet );
```

### Return Value

Always S\_OK.

### Parameters

*pvData*

[out] The buffer that holds the returned data. This value is actually an array of String that holds the preset position list.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

**Remarks**

The returned pvData is an array of Bytes. You could get the size of the image by testing the size of the array.

---

## GetSnapshot

Get the current decoded picture frame.

### Syntax

```
HRESULT GetSnapshot (
    EPixelFormat eFormat,
    Variant *pvData,
    Variant *pvInfo,
    Long *plRet
);
```

### Return Value

Always S\_OK.

### Parameters

*eFormat*

[in] The format of image caller needs. Currently, only BMP is supported.

*plData*

[out] The buffer that holds the returned data. This value is actually an array of Byte that holds the image data.

*pvInfo*

[out] Lists the information for the image. It's an array of Variant.

0: Width (Long)

1: Height (Long)

2: Quality (String) (**N/A, it's now empty string**)

3: Color / Mono (String) (**N/A, it's now empty string**)

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---



---

## GetSystemStatus (*N/A*)

Get the current system status on server.

### Syntax

```
HRESULT GetSystemStatus (    Variant *pvData,  
                           Long *plRet    );
```

### Return Value

Always S\_OK.

### Parameters

*pvData*

[out] The buffer that holds the returned data. This is an array of String:

0: Connection number

1: Width of image

2: Height of image

3 to (N + 2): Client IP addresses. N is the number of current connected client.

*plRet*

[out] the return code of the function. 0 means success, others mean failed

### Remarks

This function works in blocking mode. It will not return until the data got or HTTP time out occurred.

---

## GetUartData

Get the data on the UART interface of server.

### Syntax

```
HRESULT GetUartData (           String *pstrData,  
                                Long IReadLen,  
                                Long ITimeout,  
                                Long *pIRet           );
```

### Return Value

Always S\_OK.

### Parameters

*pstrData*

[in] The buffer that holds the returned data. This is a string that hold the hex string of the data read from COM port on Visual Server..

*IReadLen*

[in] This is the length user want. This value can't not be greater than 128.

*ITimeout*

[in, defaultval(30000)] This value is the timeout value for the reading action. This value could be skipped under VB.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This function works in blocking mode. It will not return until the data got or HTTP timeout occurred.

---

---

## GetUartDataBinary

Get the data on the UART interface of server.

### Syntax

```
HRESULT GetUartDataBinary (    Variant *pvData,  
                               Long IReadLen,  
                               Long ITimeout,  
                               Long *pIRet    );
```

### Return Value

Always S\_OK.

### Parameters

*pvData*

[out] The buffer that holds the returned data. This is an array of Byte. VC users need to free the memory by calling VariantClear() function.

*IReadLen*

[in] This is the length user want. This value can't not be greater than 128.

*ITimeout*

[in, defaultval(30000)] This value is the timeout value for the reading action. This value could be skipped under VB.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This function works in blocking mode. It will not return until the data got or HTTP timeout occurred.

---

## HttpCommand

Get the data on the UART interface of server.

### Syntax

```
HRESULT GetUartData ( String strUrlCommand,  
                      Boolean bPost,  
                      Boolean bReadData,  
                      String *strReadData,  
                      Long *plRet );
```

### Return Value

Always S\_OK.

### Parameters

*strUrlCommand*

[in] The URL command to be sent to server.

*bPost*

[in] Is the command to be sent as POST or GET. True to be sent by POST.  
False to be sent by GET.

*bReadData*

[in] Should the control try to get the returned page data? If this argument is set to be False, the following argument will be ignored.

*strReadData*

[out] The retrieved data from server. The maximum data length this command could be handled is 20K. All data beyond this range will be lost.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This function works in blocking mode. It will not return until the data got or HTTP timeout occurred.

---

---

## RecallPtzPosition

Move the camera to some preset position.

### Syntax

```
HRESULT RecallPtzPosition (      String strPosition,  
                                Long ICam,  
                                Long *pIRet      );
```

### Return Value

Always S\_OK.

### Parameters

*strPosition*

[in] This is the name of the position to be set.

*ICam*

[in, defaultval(1)] This is the camera index. Not used for connecting to Visual Servers that contain only one camera.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

---

## RepairDatabase

Repair the database when it could not opened correctly.

### Syntax

```
HRESULT RepairDatabase (      String strDatabase,  
                             Long  *pIRet      );
```

### Return Value

Always S\_OK.

### Parameters

*strDatabase*

[in] This is the full path name of the database path.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This function will create a new thread to repair the database, and once the repair finish, an event [OnRecordStatus](#) will be callback with eStatusDBRepairFinish status code, and a True or False for IParam. If the control is closing before the repair finish, the thread will be terminated and the database will be left in a inconsistent state (It's ok to call repair again to turn the database again in consistent state).

**Application must be careful not to call repair database for the same database in more than one control at the same time. The result will be unpredicted.**

If a database is open successfully, the call to this method will be failed with error code VS3ERR\_DB\_DATABASE\_INITIALED. This is to prevent repairing a database that is under use.

When a control is repairing database, any call to database related functions

---

---

will be failed with error code VS3ERR\_DB\_REPAIRING.

If a database could be opened and this method is called, an error code VS3ERR\_DB\_DONT\_NEED\_REPAIR is returned.

---

---

## RepairLocation

Repair the location when it could not opened correctly.

### Syntax

```
HRESULT RepairLocation (    String strLocation,  
                           Long *plRet                );
```

### Return Value

Always S\_OK.

### Parameters

*strLocation*

[in] This is the location name.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This function will create a new thread to repair the location, and once the repair finish, an event [OnRecordStatus](#) will be callback with eStatusLocRepairFinish status code, and a True or False for IParam. If the control is closing before the repair finish, the thread will be terminated and the database will be left in a inconsistent state (It's ok to call repair again to turn the database again in consistent state).

Application must be careful not to call repair location for the same location in more than one control at the same time. The result will be unpredicted. Usually, if one control maps to one location, this won't be an issue.

Before calling this method, the database must be opened first. If a location is open successfully, the call to this method will be failed with error code VS3ERR\_DB\_LOCATION\_OPENED. This is to prevent repairing a location that is under use.

---



---

When a control is repairing location, any call to database related functions will be failed with error code VS3ERR\_DB\_REPAIRING.

If a location could be opened and this method is called, an error code VS3ERR\_DB\_DONT\_NEED\_REPAIR is returned.

---

## RestoreControlHandle

Restore the window handle for drawing to the control itself.

### Syntax

```
HRESULT RestoreControlHandle ( Long *pIRet );
```

### Return Value

Always S\_OK.

### Parameters

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

---

## SavePresetPosition

Save the current position of the camera as a preset setting on server.

### Syntax

```
HRESULT SavePresetPosition (    String strPosition,  
                                Long ICam,  
                                Long *pIRet        );
```

### Return Value

Always S\_OK.

### Parameters

*strPosition*

[in] This is the name of the position to be set.

*ICam*

[in, defaultval(1)] This is the camera index. Not used for connecting to Visual Servers that contain only one camera.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## SaveSendMail

Save the current frame of image to a file specified. And then invoke the default mailer program to let users send out the image to remote users.

### Syntax

```
HRESULT SaveSendMail ( EPictureFormat eFormat,  
                        String strSendto,  
                        );
```

### Return Value

Always S\_OK.

### Parameters

*eFormat*

[in] The format of image caller needs. Currently, only BMP is supported.

*strSendTo*

[in] This is the 'send to' field in the mailer. It's a default value, the value could be changed after the mailer program open.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

---

## SaveSnapshot

Save the current frame of image to a file specified.

### Syntax

```
HRESULT SaveSnapshot ( EPictureFormat eFormat,  
                        String strFileName,  
                        Long *plRet );
```

### Return Value

Always S\_OK.

### Parameters

*eFormat*

[in] The format of image caller needs. Currently, only BMP is supported.

*strFileName*

[in] This is the name of the file used to save the image.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## SendCameraControl

Send server the camera control command.

### Syntax

```
HRESULT SendCameraControl (    String strCommand,  
                             Long lTimeout  
                             Long *plRet    );
```

### Return Value

Always S\_OK.

### Parameters

*strCommand*

[in] This is the command that supported by the server now:  
right, left, up, down, home, tele, wide, near(*N/A*), far(*N/A*), auto(*N/A*).

*lTimeout*

[in, defaultval(30000)] This is the timeout value for the first five command listed above. For VB users It could also be skipped to use the default value.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

---

## SendCameraControlMap

Send server the camera control command with relative or absolute coordinate.

### Syntax

```
HRESULT SendCameraControlMap ( Long IX,  
                                Long IY,  
                                ECoordinateType eType,  
                                Long *pIRet );
```

### Return Value

Always S\_OK.

### Parameters

*IX*

[in] The x coordinate related to the upper-left corner of the control or the x amount of movement related to current position.

*IY*

[in, optional] The y coordinate related to the upper-left corner of the control or the y amount of movement related to current position.

*eType*

[in, defaultval(1)] Indicate the type of the movement. Could be Relative or Absolute. If this value is skipped, the absolute type would be chosen automatically.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This method is only effective if the server is running with new firmware that supports click on image.

---

## SendCameraControlSpeed

Send server the camera control command to control the camera movement speed.

### Syntax

```
HRESULT SendCameraControlSpeed ( ESpeedType eType,  
                                Long ISpeed,  
                                Long *pIRet );
```

### Return Value

Always S\_OK.

### Parameters

*eType*

[in] The speed type, either Tilt speed (2) or Pan speed (1).

*ISpeed*

[in] The speed value, the range should be 5~-5 include 0.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

This method is only effective if the server is running with new firmware that supports pan/tilt speed setting.

---



---

## SendDigitalOut

Send server the digital output to certain port.

### Syntax

```
HRESULT SendDigitalOut (    Long IPort,  
                           Long IValue,  
                           Long *pIRet    );
```

### Return Value

Always S\_OK.

### Parameters

*IPort*

[in] Specify the port index of the digital output device to set data. Port index starts from 0. It is not used if the server contains only one port.

*IValue*

[in] 0 for low level signal, 1 for high level signal. Other values are reserved for future uses.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## SendUartCommand

Send server the command to certain COM port.

### Syntax

```
HRESULT SendUartCommand (    String strCommand,  
                             Long lPort,  
                             Long *plRet    );
```

### Return Value

Always S\_OK.

### Parameters

*strCommand*

[in] This is the command to send to UART on server. It's hex string converted from the binary command.

*lPort*

[in, defaultval(1)] Specify the COM port index. It could be skipped for VB users. This parameter is not used if the server contains only one COM port.

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

---

## SendUartCommandBinary

Send server the command to certain COM port.

### Syntax

```
HRESULT SendUartCommandBinary ( Variant vCommand,  
                                Long IPort,  
                                Long *pIRet );
```

### Return Value

Always S\_OK.

### Parameters

*vCommand*

[in] This is the command to send to UART on server. It's an array of Byte.

*IPort*

[in, defaultval(1)] Specify the COM port index. It could be skipped for VB users.

This parameter is not used if the server contains only one COM port.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## SetDatabasePath

Set the database path used by this control. This database is used only for recording.

### Syntax

```
HRESULT SetDatabasePath (      String strPath,  
                               Boolean bAutoCreate,  
                               Long *plRet           );
```

### Return Value

Always S\_OK.

### Parameters

*strPath*

[in] This is the path points to when the database exists.

*bAutoCreate*

[in] Should the control create the database if it cannot find any media database at the path you specified?

*plRet*

[out] the return code of the function. 0 means success, others mean failed.

---

---

## SetLocation

Set the location for storing media data.

### Syntax

```
HRESULT SetLocation (           String strLocation,  
                                Boolean bAutoCreate,  
                                Long *pIRet           );
```

### Return Value

Always S\_OK.

### Parameters

*strLocation*

[in] This is the subdirectory under the database to store data.

*bAutoCreate*

[in] Should the control create the location if it can't find the matched location under current database?

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## StartMediaRecord

Set the [MediaRecord](#) flag to true. This function will check if all the setting is correct for recording, for example, the [SetDatabasePath](#) and [SetLocation](#) must be called before calling this method. This method is the same as calling [StartMediaRecordEx](#) with event record flag set to False.

### Syntax

```
HRESULT StartMediaRecord (    EMediaType eMediaType,  
                             Long *pIRet    );
```

### Return Value

Always S\_OK.

### Parameters

*eMediaType*

[in] Which media to record to the database? Currently, this parameter is not implemented and reserved for future.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

**The recording will start after the control connects to server. So the return of this method doesn't always mean that the recording starts (It marks the recording flag internally). If you call this method during connecting mode, the recording starts immediately.**

To let users knows that the control is recording. The caption of the control would turn RED when recording, and turn back to normal color after recording stops.

---

---

## StartMediaRecordEx

Set the [MediaRecord](#) flag to true. This function will check if all the setting is correct for recording, for example, the [SetDatabasePath](#) and [SetLocation](#) must be called before calling this method. This method is to replace [StartMediaRecord](#). Users could specify if the recording is normal continuous recording or event recording.

### Syntax

```
HRESULT StartMediaRecordEx (    EMediaType eMediaType,  
                                Boolean bEventRecord,    );  
                                Long *pIRet                );
```

### Return Value

Always S\_OK.

### Parameters

*eMediaType*

[in] Which media to record to the database? Currently, this parameter is not implemented and reserved for future.

*bEventRecord*

[in] Whether the recording is normal continuous recording or a event driven recording. For more detail about event recording, please see the remarks section below.

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

### Remarks

**The recording will start after the control connects to server. So the return of this method doesn't always mean that the recording starts (It marks the recording flag internally). If you call this method during connecting mode, the recording starts immediately.**

---

---

Event recording means the control will start recording when events are triggered. Users could select which kind of events to start recording by setting [EventTypes](#). To prevent the recording contains no important information because the time when event triggered the video is not I frame, the control will pre-buffer a sequence of video frame. The length of time to pre-buffer could be set by [PreEventTime](#). The recording will stop automatically after a period of time. The period could be adjusted by setting [PostEventTime](#) (For DI high or DI low, the recording continues until the DI state changes). If events are triggered continuously, the recording might last for a long time. This is to prevent losing important information because event might happen between the time recording stopped and the next I frame comes. In such case, that event would be lost (not recorded).

To let users knows that the control is recording. The caption of the control would turn RED when recording, and turn back to normal color after recording stops.

---



---

## StopMediaRecord

Set the recording flag of the control to false.

### Syntax

```
HRESULT StopMediaRecord (        Long *pIRet                    );
```

### Return Value

Always S\_OK.

### Parameters

*pIRet*

[out] the return code of the function. 0 means success, others mean failed.

---

## 3.3 Events

This control supports connection point. With this mechanism, control owner could receive certain events when certain condition happens. To receive these events, VC users should implement the event-sinking interface. Readers could find the example of how to implement the event-sinking interface using MFC in the sample codes. For those that don't use MFC, please search on the Internet for the ATL implementation of sinking target. VB users could easily implement the events by click on

**Procedures/Events Box** to insert the events.

---

---

## OnClick

The control fires this event whenever user clicks on the control by using mouse pointer and the [ClickEventHandler](#) is set to HandleSelf or HandleSelfSendEvent.

### Syntax

```
HRESULT OnClick (    long IX,  
                   long IY                );
```

### Return Value

Please always return S\_OK.

### Parameters

*IX*

[in] The x coordinate related to upper-left corner of the control where user clicked mouse.

*IY*

[in] The y coordinate related to upper-left corner of the control where user clicked mouse.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally.

---

## OnConnectionBroken

The control fires this event whenever the connection is broken by some reason other than user stop it.

### Syntax

```
HRESULT OnConnectionBroken ( EConnectionType eConnType );
```

### Return Value

Please always return S\_OK.

### Parameters

*eConnType*

[in] This parameter indicates the connection type of connection that is broken. Two values are available now: connAudio, connVideo.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally.

---

---

## OnConnectionOK

Sent whenever the control connects to the Visual Server successfully.

### Syntax

```
HRESULT OnConnectionOK ( EConnectionType eConnType, );
```

### Return Value

Please always return S\_OK.

### Parameters

*eConnType*

[in] This parameter indicates the connection type of connection that is broken. Two values are available now: connAudio, connVideo.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally.

---

## OnDIDOAlert

The control fires this event whenever it gets the window alert event from video stream.

### Syntax

```
HRESULT OnDIDOAlert (           Long IChangeFlag,   );  
                               Long IDIDOValue
```

### Return Value

Please always return S\_OK.

### Parameters

#### *IChangeFlag*

[in] This parameter contains DI & DO change flag, the low word saves DI change flag, and the high word contains DO change flag. If bit 0 of low word is 1 means DI changes, 0 means does not change. If bit 0 of high word is 1 mean DO change (might be changed by other client), 0 means it does not change.

#### *IDODOValue*

[in] This parameter contains DI & DO current value. The low word saves DI value, and the high word contains DO value. If bit 0 of low word is 1 means DI is high, 0 means DI is low. If bit 0 of high word is 1 mean DO is high, 0 means DO is low.

### Remarks

When implement this event function, the function should be declared as void.  
The return value here is used by OLE library internally.

---

---

## OnMDAlert

The control fires this event whenever it gets the window alert event from video stream.

### Syntax

```
HRESULT OnMDAlert ( Variant vStatus );
```

### Return Value

Please always return S\_OK.

### Parameters

*vStatus*

[in, ref] This parameter contains a two-dimension array that holds the information for the three motion detection windows. *vStatus*[*n*][0] is a Boolean value that identifies if alert happens for the corresponding window. *vStatus*[*n*][1] is a Long value that hold the percentage of alert.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally. **Note: for the way to retrieve the array from the variant in various language, please refer to the sample code.**

---

## OnNewAudioPiece

The control fires this event whenever it comes the new audio piece from the connection.

### Syntax

```
HRESULT OnNewAudioPiece (     Variant vPiece     );
```

### Return Value

Please always return S\_OK.

### Parameters

*vPiece*

[in, ref] This parameter contains the new arrival audio data. It's an array of Byte.

### Remarks

When implement this event function, the function should be declared as void.  
The return value here is used by OLE library internally.

---



---

## OnNewImage

The control fires this event whenever a new image frame is received from server.

### Syntax

```
HRESULT OnNewImage (    Boolean bSignal
                        DWORD dwDI                );
```

### Return Value

Please always return S\_OK.

### Parameters

*bSignal*

[in] Indicates if there are signal for the new image. It's only available for new version firmware

*dwDI*(**N/A**)

[in] Indicates the DI status, each bit represents on digital input port. Bit on for high, 0 for low.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally.

---

## OnNewVideo

The control fires this event whenever a new image frame is received from server and users request to get the decoded video.

### Syntax

```
HRESULT OnNewImage (    Boolean bSignal,  
                        Boolean bDecodedImg  
                        Variant vPiece                );
```

### Return Value

Please always return S\_OK.

### Parameters

#### *bSignal*

[in] Indicates if there are signal for the new image. It's only available for new version firmware

#### *bDecodedImg*

[in] Indicates if the notification contains data. Users should use the *vPiece* only if this flag is set to True.

#### *vPiece*

[in, ref] This parameter contains the new arrival video data. It's an array of Byte.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally.

---

---

## OnRecordStatus

The control fires this event whenever error happens during recording.

### Syntax

```
HRESULT OnRecordStatus ( Long IStatus  
                        Long IParam );
```

### Return Value

Please always return S\_OK.

### Parameters

*IStatus*

[in] Indicates the status code for this notification, currently, only 1 is available. Which means the disk is full, no space left for recording. Note, you have to handle the disk full event by stopping the recording, or you will get this event each time when network packets arrive.

*IParam*

[in] This argument is now not used. Reserved for future.

### Remarks

When implement this event function, the function should be declared as void. The return value here is used by OLE library internally.

---

## 3.4 Error Code List

The following is the error list for the control. The error code is returned by the last parameter of each method.

Code	Name	Meaning
1001	VS3ERR_CREATE_CONNECTION	Error when create connection.
1002	VS3ERR_HTTP_BLK_ERR	Error when doing http blocking operation, such as GetPtzPosition.
1003	VS3ERR_CONNECT	Error when connecting to server. It is returned when calling Connect().
1101	VS3ERR_DB_PATH_INCORRECT	The database path is not correct. Either it doesn't exist or it points to a file rather than a path.
1102	VS3ERR_DB_NOT_EXIST	Database doesn't exist. When giving database path, caller could ask to create the database automatically or just open existing database. This error code is returned if callers want to open existing database, but it is not there.
1103	VS3ERR_DB_CR_FAILED	Failed to create database. Maybe the disk is full or because of permission denied.
1104	VS3ERR_DB_NOT_INIT	Database is not initialized. This is returned when calling SetLocation.
1105	VS3ERR_DB_LOC_NOTFOUND	The specified location is not found. This is returned when callers don't want the control to create the location automatically and the location doesn't exist.
1106	VS3ERR_DB_LOC_CR_FAILED	Create location failed. The possible reasons are disk full or users have no permission to create directory or create file on the target directory.
1107	VS3ERR_DB_LOC_OTHER_ERR	Other location related error.
1110	VS3ERR_DB_LOC_NOT_INIT	The location is not initialized before calling StartMediaRecord
1117	VS3ERR_DB_DONT_NEED_REPAIR	Database does not need to be repaired. It's healthy.
1118	VS3ERR_DB_REPAIRING	Database is under repairing, no other

		database operation is permitted.
1119	VS3ERR_DB_LOCATION_OPENED	Location is opened. In such case, repairing database or repairing location are both not permitted.
1120	VS3ERR_DB_DATABASE_INITIAL D	Database is opened. In such case, repairing database is not permitted.
1201-1 214		Internal component Error
1301-1 307		Memory related error
1401	VS3ERR_URL_MISS_DI	URL for retrieving DI status is empty.
1402	VS3ERR_DI_INCORRECT_FORM	The returned DI status page from server is not the format that we know. It's possible that the server is not in the list that this control supports.
1403	VS3ERR_PRESET_INCORRECT_F ORM	The returned Preset location page from server is not the format that we know. It's possible that the server is not in the list that this control supports.
1404	VS3ERR_SNAPSHOT_FMT_UNSU PP	The snapshot output format is not in the supported list.
1405	VS3ERR_SNAPSHOT_FAILED	Snapshot failed.
1406	VS3ERR_URL_MISS_UART	URL for UART read/write is empty.
1407	VS3ERR_URL_MISS_PTZURL	URL for PTZ control is empty.
1408	VS3ERR_EXCEED_MAX_LEN	The given buffer length exceeds the maximum COM port read/write buffer length (128)
1409	VS3ERR_POINT_NOTIN_CTRL	The coordinate of the given point is not within control client area.
1410	VS3ERR_URL_MISS_DO	URL for DO setting is empty.
1411	VS3ERR_PARAM_INCORRECT	The vData parameter for SendUartCommandBinary() is not a byte array
1412	VS3ERR_URL_MISS_PRESET	URL for preset page is empty.
1413	VS3ERR_URL_MISS_VAM	URL for A/V stream is empty.
1415	VS3ERR_IPHOSTNAME_MISS	Does not give any IP or hostname info before calling Connect
1416	VS3ERR_SNAPSHOT_NOT_DECO DE	Snapshot is requested but the DecodeAV property is set to False.

1417	VS3ERR_NOT_CONNECTED	Snapshot could not be got if not connect to server
1501-1 503		File system error
3001		No match items

---

---

---

# Chapter 4

## Profile



This chapter contains code size.

---

## 4.1 Code Size

The code size for the control and the signed cab file is as followings:,

File	Size (Bytes)
VADecoder.dll	567,296
VaCtrl.cab	301,157