MediaPlayback Control

Version 2.2.0.7

2007/11/22

TABLE of CONTENTS

Chapter 1

Overview

1.1 Introduction

This document describes the properties and methods supported by the MediaPlayBack ActiveX control. MediaPlayBack ActiveX control, in this release, supports all series model products.

Getting Started with MediaPlay Control

The main function of MediaPlayBack 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#.

File based V.S. database based mode

The control now supports two modes: One is file-based mode, and the other is database-based mode. The latter is to open a whole location of a database and play the files continuously. The file-based mode is new for the control. It could play any single hgd file even it is located in remote http server. With the file-based mode, it's now possible to play histogram data remotely.

Installing the ActiveX control

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

1. Download the installation file MPlayback.zip and unpack it. If you get the distribution disc, just find the MPlayback.exe under lib subdirectory.
2. Run the MPlayback.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.

File Structure

FILE	DESCRIPTON
doc\ MediaPlayBack Control.doc	This manual

lib\MPlayback.exe	The installation file for the control itself and the cab file that signs the control.
sample\VbTest	Live video sample code for VB

1.2 Release note

Version 2.2.0.7

- System Requirements
  Software:
  1. Windows 98/ME/2000/XP
- Modify
  - Add Deinterlace to do deinterlacing when decode video stream.

Version 2.2.0.5

- System Requirements
  Software:
  1. Windows 98/ME/2000/XP
- Fixed Bugs
  - SetEventTypes would not do query again to get event numbers and interval number, so it will return very fast
  - CountIntervalEventThreadProc would set interval number event right after interval queried ok. So it would not block the drawing thread
  - GetMatchedEvent would not query number first, it queries the data directly so that it would return faster
  - GetCurrentEventInfo would not query number first, it queries the data directly so that it would return faster
  - DoQueryInformation would not wait until interval number ready, so it would not block

SetEventTypes method
- Could abort Query function for event or interval number to prevent error
- AVI conversion will continue even if any image frame damaged or lost in Database
- Solve the MediaDBPlayback seek problem in database

Version 2.2.0.2

- System Requirements
  Software:
  2. Windows 98/ME/2000/XP
- Changes
  - Add CurrentTimeSec to let application knows where the media is played.
  - Add SetEndOfInterval and OnRequestNextPlayFileName to let application set the file mode type. And could let application to play in file mode continuously.
- Fixed Bugs
  - Snapshot for BMP would get width and height information with 0. Fixed
  - The sync for playback is incorrect. Fixed
  - Notify for the playback time is not exactly, now fixed
  - Slider used in IE does not work properly, fixed
  - Video only sync has problem, fixed
  -

Version 2.2.0.0

- System Requirements
  Software:
  3. Windows 98/ME/2000/XP
- Changes
  - Add ForceNonYUV property.

- Add NextFrame method.
- Add GetSnapshot, SaveSendMail, and SaveSnapshot
- Add DBHierarchy, Deblocking, DisplayMotionFrame, DisplayPeriod, JpegQuality, PlayMute, PlayVolume properties
- Use new underlying database module. So when repair, the repaired database would be new format. But control could still read previous database.

- Fixed Bugs
  - For http remote playback, now the http port could be assigned after the IP address with ':' as seperator.
  - For AVI conversion, MS WMV9 does not work in previous version, now it works.
  - Many audio codecs in AVI conversion would report incorrect media length or the AVI would be played back in a strange way, now fixed. For audio codecs that could not be supported the audio track would not be put in the generated AVI files (video only).
  - The non-triggered motion frame is removed to let control consistent with VitaminCtrl
  - Fix memory leak problem for AVI conversion
  - Some 7000 file-base hgds could not be played successfully. Fixed
  -

Version 2.0.0.0

- System Requirements
  Software:
  4. Windows 98/ME/2000/XP
- Changes
  - The Location property is renamed to DBLocation. It avoids the confliction of default property in C# program.

- The Histogram map is now using a separate thread to paint to avoid UI blocking.
- The time display is moved to right handle side of the caption to conform to other ActiveX control.
- Support file base and remote file (hgd) only playback.
- Add ActiveBuffering, AutoPlay, PlayFileName properties
- Add ReleaseBeforeClose method.

- Fixed Bugs
  -

Version 1.2.0.3

- System Requirements
  Software:
  5. Windows 98/ME/2000/XP
- Changes
  - Implement the IStreamStorage interface so that the control could be updated without recompile the program.
  - Support multiple monitors now.
- Fixed Bugs
  - Stop play would block, fixed.
  - Fix Ignore border bug in last version.
  - Fix force GDI bug.
  - When pause, click on slider or histogram is now of no use to prevent dead-lock.

Version 1.2.0.1

- System Requirements
  Software:
  6. Windows 98/ME/2000/XP
- Changes
  - Add DisplayTimeFormat and UserDateFormat to let all value in UI is conformed to the setting users

like.
- Could show slider and map in IE. The position of the slider and map are fixed.
- Add AVI exporting support (properties starts with AVI and functions name contain AVI)
- Add caption/border color setting and drawing code
- Add Repair function for location (RepairLocation) and database (RepairDatabase)
- Add Close database (CloseDatabase)/location (CloseLocation) functions
- Add delete location function DeleteLocation
- Add Test location function TestLocation

- Fixed Bugs
  - Fire_OnPlaybackStatus is called from UI thread to prevent crashing if AP call UI subroutine.
  - Fix resource leak when thread ends.
  - Slider update could sometimes block, fixed.
  - Solve blocking problem in playback thread.
  - When drawing event, the left part should be adjust if the event is covered the selected time period.
  - Avoid flash when playing point moves by clicking on map or slider.
  - Fixed a bug that the second time segment would have no DI low event.
  - fixed a bug that the time interval within selected period (the interval is smaller) would lost DI when redraw.
  - The invert function for Map does not work when X < 0, fixed.
  - Let it could select period when X < 0 or X > right.
  - When coordinate of display area is less than 0 (either X or Y) the video shown strangely. Fixed.
  - Fix the concurrent database location access problem (with other recording program)
  - SetDatabasePath and SetLocation will close current database or location and then

- Open the new database, location, if the new one is opened fail, no rollback will happen the database state will remain as un-opened.

Version 1.1.0.1

- System Requirements
  Software:
  7. Windows 98/ME/2000/XP
- Changes
  - Add ForceGDI property
  - Add IgnoreBorder property
  - Add IgnoreCaption Property
  - OnNotifyTime event interface is changed slightly to make it works under script language.
  -
- Fixed Bugs
  - When location contains no data, error code should be returned rather than raising COM exception.
  - OnNotifyTime for playtime had problem in last version. VB sample would crash after handling this event. Fixed

Version 1.0.0.14

- System Requirements
  Software:
  8. Windows 98/ME/2000/XP
- Changes
- Fixed Bugs
  - Could receive array from vb script now.
  - Daylight saving conversion is changed to resolve the bug that –1 is shown when hour is 0.
  - PlayFromTime didn't work correct in last version. Fixed.
  - Fix the MatchEvenList caused OLE runtime error.

Version 1.0.0.7

- System Requirements
  Software:
  9. Windows 98/ME/2000/XP
- Changes
- Fixed Bugs
  - Frames with packet lost won't hang program now.
  - Change screen resolution during playing will not crash now.
  - Installation will register the control automatically.

Version 1.0.0.6

- System Requirements
  Software:
  10. Windows 98/ME/2000/XP
- Changes
  - Use GDI if the display card does not support direct draw.
- Fixed Bugs
  - Audio quality adjustment
- Known Bugs
  - The control will crash when changing resolution during playin

Version 1.0.0.0

- System Requirements
  Software:
  11. Windows 98/ME/2000/XP
- Features
  - Provide build-in slider and media map to help control the media playback.
  - Support automatic media/codec switching for database media playback.
- Notes

Chapter 2

PROGRAMMER'S GUIDE

2.1 Using control within 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 **MediaPlayBack class** in the list box.
3. Click **Insert** button to close the dialog, and the MediaPlayBack object will now appear in the toolbox.
4. Now you can drag the MediaPlayBack 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 MediaPlayBack

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_MediaPlayback**.

Set Database Path and Location to get ready to play media back

The basic operation is to set database path and choose one location from it to retrieve media and play them on screen. When using the control, you must call SetDatabasePath and SetLocation functions to get the control ready to play media. Optionally, you could list the locations contained in one database. The function GetLocationList is for this purpose.

Add the following sample code in your **OnInitDialog** function so that the program will start playing the media in the database. We omit the error handling here to be simplified. “Channel1” is assumed to be one location in your database. You could imagine that the location stores media data from one camera in your site.

```
long lRet;
m_MediaPlayBack.SetDatabasePath(“c:\\MediaDB”, &lRet);
m_MediaPlayBack.SetLocation(“Channel1”, &lRet);
m_MediaPlayBack.Playback(&lRet)
```

Retrieve more information about a location

In one location, there should be many pieces of time you record data. We call these pieces as time interval. One interval means a continuous time frame you store data. Once you stop recording, the time interval breaks. So next time when you record, the database will save the media to a new time interval.

Some time if you record some pieces of data and reset the clock time on server. Then you record again. Due to the time adjustment, there maybe some time intervals duplicate. For example, you record an interval between 2003/4/10 09:00:00 to 2003/4/10 10:00:00. And then you adjust the server time back to 2003/4/10 09:30:00 and start recording again. When happens to the database? It will not overwrite the data piece after 09:30:00. Rather, it will create another chunk of time intervals to store these new data. We call the chunk of time interval as time segment. So in a location, it may contain 1 to many time segments, and each time segment contains many time intervals. If the database generation engine doesn’t detect any time interval conflict, one location will usually contain only one time segment.

So you could list the time segments of a location, and assign the current time segment index. This index is used when you do any operation to this location. If you don’t specify segment index, 0 will be the default value every time when you change location. The function used to list segment is GetSegmentList, and the property used to set/get current segment is CurrentSegment. You could refer to the chapter 3 for more details.

After choosing the segment, you could advance to browse the contain of the segment. That is the time intervals contained in this segment. GetTimeIntervalList is for this purpose.

Playback Method and Speed

There are three play methods for this control: full range, selected period, and matched event. Full range will list all the timer intervals contained in a time segment. The selected period allows you to specify a subset of time frame to play the media. And the event matched method cares only events happen during the time period you specify. The property PlaybackMethod is for this purpose. You could change the method at any time, but this could affect the playing behavior of the control.

When playback media from database, you could play in fast forward mode or see slow motion just like traditional VCR. But the maximum speed of fast forward mode depends on the CPU power. Currently we support 32 different speeds for playback including normal speed. The fastest speed is 16X, but only very high end PC could reach this speed. In fast forward mode, the control will skip some frame for better performance. When not playing in normal speed, the audio will be disabled. The property PlaybackSpeed is used to control the speed.

Search Events

Events are some special signal sent out by server to notify that something happens, such as motion detection alert and digital input alert. You could search in the database for the event that you are interested in by specifying event types. The method SetEventTypes is for this purpose.

After you set the filter to search event, you could retrieve the matched event list. The list contains an array of event information. You get the array by calling GetMatchedEventList.

If the playback method is matched event mode, you could choose an event to play, this event is called the current event. The property CurrentEvent is the index of current playing event.

Slider and Media map

The control supports Slider and Media map. These accessories could help users to better understand current playing point and the time intervals and events contains in current select period (or full time range). Besides the display functionality, users could

also click on these two accessories to jump to the point they want or change playing period. More details could be found in chapter 3. Basically, Slider and media map are created by this control but it could be put anywhere other than the client area of the control. That means, you could assign a new parent for these accessories and move them to anywhere you want. The properties SliderVisible, MediaMapVisible control the show or hide of them, and the SetSliderParam and SetMediaMapParam set the parent, position, and size of these accessories.

For these two accessories, you could freely assign the colors for various parts. Please refer to the properties for more details.

For IE users, the control will put the slider and map right below the display area. The height of slider and map are fixed. For slider the constant height is 50, and the map is 60 pixels. The widths of these two accessories are the same as the width of the ActiveX control. It’s possible to turn on or turn off the accessories by setting properties SliderVisible and MediaMapVisible.

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.

EAVINotifyStatusCode

List Member

Name	Value	Description
eConversionStopped	1	The AVI conversion has been stopped either aborted by calling StopAVIConversion or the conversion has been finished for all the media data. For this status, the lParam1 contains the reason that conversion stop. 1 means users interrupted (calling StopAVIConversion). 0 means conversion finished. The lParam2 contains the total size of files generated in KB.
eConversionError	2	The AVI conversion has been stopped with error. For this status, the lParam1 contains the error code. The lParam2 contains the total size of files generated in KB if any.

Description

This enumeration is used for OnAVINotify event.

EDBEventType

List Member

Name	Value	Description
eMDAlertWin1	1	Event for motion detection window 1 alert
eMDAlertWin2	2	Event for motion detection window 2 alert
eMDAlertWin3	4	Event for motion detection window 3 alert
eDILow1	256	Event for DI 1 is low
eDILow2	512	Event for DI 2 is low
eDILow3	1024	Event for DI 3 is low
eDILow4	2048	Event for DI 4 is low
eDIHigh1	65536	Event for DI 1 is high
eDIHigh2	131072	Event for DI 2 is high
eDIHigh3	262144	Event for DI 3 is high
eDIHigh4	524288	Event for DI 4 is high

Description

This enumeration is used for EventTypes property and SetEventTypes method.

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.

ENotifyStatusCode

List Member

Name	Value	Description
eStartPlay	1	The control start to play the media. This is current not used.
eStopPlay	2	The playing was stopped.
eEventIndexChanged	3	The event index is changed. This may be caused by user click the event on the media map.
ePlaybackMethodChanged	4	The playback method was changed due to some settings changed. For example, when location changed, the playback method is changed to full range mode automatically.
eLimitedPeriodChanged	5	The selected period is changed. This happens when users select a new area on the media map.
eStatusDBRepairFinish	6	The database repair is finished. The lParam of the event means whether the repair success or not. Nonzero means success, 0 means not.
eLimitedPeriodChanged	7	The location repair is finished. The lParam of the event means whether the repair success or not. Nonzero means success, 0 means not.
eStatusNetInitFailed	8	When run in remote file base mode, the network connection could not be established.
eStatusFBDBInitFailed	9	The specified file to be played back is not a correct “hgd” file.
eStatusFBDownloadDone	10	The remote file is downloaded OK
eStatusNetConnectFailed	11	Connect to remote http server failed when retrieving the remote hgd file.

Description

This enumeration is used when firing OnPlaybackStatus event.

ENotifyTimeType

List Member

Name	Value	Description
ePlayTime	1	The playing time point changed, with getting this event, you could update the playing time show on screen.
eNoMatchAtTime	2	When users click on the slider or media map, and the clicked time contains no media, this event will be fired. You could show message to alert user no matched media with response to this event.
eNoMatchAtTimePeriod	3	When users mark an area on media map and release the mouse, the control will check if there are any media during this time frame. If not, this event will be fired. You could show error message with response to this event.

Description

This enumeration is used when firing OnNotifyTime event.

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.
EPicFmtIYUV	5	Planar format. All Ys follow by all Us and then all Vs
EPicFmtYV12	6	Planar format. All Ys follow by all Vs and then all Us

Description

This enumeration is used when get image from control. For snapshot, the jpeg mode might need to encode the decoded data to jpeg again. The performance might be bad.

EPlaybackMethod

List Member

Name	Value	Description
eFullRange	1	The full time segment range.
eSelectedPeriod	2	Users specified time period, the range should be a subset of full range.
eMatchedEvents	3	Play only the period event happens. When playing the real piece would be about 8-10 seconds contains the time point when event happens.

Description

This enumeration is used for PlaybackMethod property and SetPlaybackMethod method.

EPlaybackSpeed

List Member

Name	Value	Description
eOneSixteenth	-16	Playback in 1/16 speed
eOneFifteenth	-15	Playback in 1/15 speed
eOneFourteenth	-14	Playback in 1/14 speed
eOneThirteenth	-13	Playback in 1/13 speed
eOneTwelfth	-12	Playback in 1/12 speed
eOneEleventh	-11	Playback in 1/11 speed
eOneTenth	-10	Playback in 1/10 speed
eOneNineth	-9	Playback in 1/9 speed
eOneEightth	-8	Playback in 1/8 speed
eOneSeventh	-7	Playback in 1/7 speed
eOneSixth	-6	Playback in 1/6 speed
eOneFifth	-5	Playback in 1/5 speed
eOneFourth	-4	Playback in 1/4 speed
eOneThird	-3	Playback in 1/3 speed
eOneSecond	-2	Playback in 1/2 speed
eNormal	1, -1	Playback in 1/1 speed
eFreeGo	0	Play in full speed, it depends on the CPU power, and normally this will not be faster than 16X since it doesn't skip frame. But when the CPU power is highly raised, this mode could be faster than 16X
eTwoTimes	2	Playback in 2X speed
eThreeTimes	3	Playback in 3X speed
eFourTimes	4	Playback in 4X speed
eFiveTimes	5	Playback in 5X speed
eSixTimes	6	Playback in 6X speed
eSevenTimes	7	Playback in 7X speed
eEightTimes	8	Playback in 8X speed
eNineTimes	9	Playback in 9X speed
eTenTimes	10	Playback in 10X speed
eElevenTimes	11	Playback in 11X speed
eTwelfTimes	12	Playback in 12X speed

eThirteenTimes	13	Playback in 13X speed
eFourteenTimes	14	Playback in 14X speed
eFifteenTimes	15	Playback in 15X speed
eSixteenTimes	16	Playback in 16X speed

Description

This enumeration is used with the PlaybackSpeed property.

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 AVIRegRoot property.

2.3 Application Sample Code

In the shipment, there are several sample codes. The following list the function for the sample code.

Vbtest

This project is the sample code written in VB 6.0. It demonstrates most of the function in this control. The functions include:

- Playback the media from a location in a database
- Playback the media from a media file (hgd)
- Display the slider and histogram map
- AVI conversion
- Misc. Play methods (full, time interval, event mode)
- Pause/Resume/Frame by frame browse
- Playback speed adjustment
- Snapshot when playback
- Misc. Options of the control controlling

Vctest

This project is the sample code written in VC++ 6.0. It demonstrates most of the function in this control. The functions include:

- Playback the media from a location in a database
- Playback the media from a media file (hgd)
- Display the slider and histogram map
- AVI conversion
- Misc. Play methods (full, time interval, event mode)
- Pause/Resume/Frame by frame browse
- Playback speed adjustment
- Snapshot when playback
- Misc. Options of the control controlling

CSharptest

This project is the sample code written in C#. It demonstrates most of the function in this control. The functions include:

- Playback the media from a location in a database
- Playback the media from a media file (hgd)
- Display the slider and histogram map
- AVI conversion
- Misc. Play methods (full, time interval, event mode)
- Pause/Resume/Frame by frame browse
- Playback speed adjustment
- Snapshot when playback
- Misc. Options of the control controlling

HTML samples

The html files are installed in the same directory as the control is. The samples include:

- Simple playback control base page, the directory is left for user to specify.
- Filebase playback, it plays the local file shipped with the package
- Remote playback, it plays a remote file, to make it works, you must change the remote file path to a reasonable server address and path.

Chapter 3

Application Program Interface Reference

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

3.1 Properties

ActiveBuffering

This property decide if the control will show buffering prompt when running in remote file based playback mode. In such case, the file will be retrieved by HTTP protocol. Due to network bandwidth limitation, the file needs to be sent in several seconds to several minutes. If the bandwidth is low the playback effect would be bad if the control play the data right after it receives them. To improve the playback effect, the control could display buffering when receiving data. These received data will be sent to playing thread after some predefined percentage reaches.

Type

Boolean

True means to enable buffering, False means not.

Attribute

R/W

Remarks

If the data file is huge, the control will use receiving bytes rather than percentage to do buffering.

AutoJumpTimeInterval

A property to control whether the control should jump to next time interval when finish playing current time interval. This would not control if the playing should be turn back when reaches the end of the time segment or the setting point of time. For circular play mode use AutoRepeat instead. This property has no effect when in event matched method.

Type

Boolean

True means to jump, False means not.

Attribute

R/W

Remarks

None.

AutoPlay

A property controls whether the control should auto start to play a file if the control is running in IE.

Type

Boolean

True means to automatically starts the playback after control created. False to stop it.

Attribute

R/W

Remarks

This property is meaningful only if the control is running under IE. This property let IE based program to run automatically with out the help of user to click on “start” button.

This property must be used together with PlayFileName property. If the file is empty, the playback will not run.

AutoRepeat

A property controls whether the control should play in circular mode. It means to go back to the head and continue playing when reach the end of the period selected or time segment. This property has no effect when playing in matched event method.

Type

Boolean

True means to repeat when reach end and False means not to repeat.

Attribute

R/W

Remarks

None

AVIFileName

This property is to be used as the output file name for AVI conversion. The value should be updated to new one when applications get OnRequestAVIFileName event. Or the file will be overwritten and hence the conversion will output incorrect number of converted AVI files.

Type

String

This is the full path name for the output AVI file name.

Attribute

R/W

Remarks

The time to set this property is when getting OnRequestAVIFileName event callback.

AVIRegRoot

This property is to specify the root key for storing and retrieving the AVI audio and video compressor settings. It's only valid when AVIRegSubKey is not empty string.

Type

ERegistryRoot

This is the root value. The possible values are list in ERegistryRoot.

Attribute

R/W

Remarks

The default value is to store the setting under "local machine". Note: If the users running this control do not have access right for the registry key, no error will be reported. The result is the settings are not saved.

AVIRegSubKey

This property is to specify the sub key under AVIRegRoot that is used to save the AVI related video and audio compressor settings.

Type

String

This is the name include registry key separators (‘\’ character).

Attribute

R/W

Remarks

It’s important to specify the correct path name for the sub key. The control will not check the correctness for the string specified but to try to open the key. If the key is opened successfully, the settings will be stored there. The default value for this string is “Software \MediaAVISetting”.

If applications are using multiple controls and each control need different setting, please set this property for each control to a different value. If these controls need the same settings, this sub key could be the same for all of them.

AVIVideoFrameRate

This property is to specify or retrieve the video frame rate used to convert A/V data in database to AVI file.

Type

Long

The permitted range for this value is from 1 to 30.

Attribute

R/W

Remarks

This frame rate could be different from the actual frame rate recorded in the database. The control will insert dummy frames or drop frames to meet this frame rate for the final AVI files. This means the video data in AVI files will not be played in fast or slow motion effect no matter what value this property is set to.

AVIVideoHeight

This property is to specify or retrieve the video height for the final AVI file. This value could be different from the video height of the actual video frame in database.

Type

Long

The value should be a multiple of 8. If not, some video compressor would have problem.

Attribute

R/W

Remarks

This property is only valid if the AVIVideoSizeByStream is set to False. If AVIVideoSizeByStream is set to be True, it's ignored. All the video frame would be stretched to meat this height if the size is different.

AVIVideoSizeByStream

This property is to determine if the video height and width is set according to the first video frame size when conversion or not.

Type

Boolean

True means to use the first video frame size as the output video size in the AVI file. All subsequent video frames would be stretched to the same size as the first frame is. False means to use the width and height specified in properties AVIVideoHeight and AVIVideoWidth.

Attribute

R/W

Remarks

When this property is set to True, AVIVideoHeight and AVIVideoWidth are ignored.

AVIVideoWidth

This property is to specify or retrieve the video width for the final AVI file. This value could be different from the video width of the actual video frame in database.

Type

Long

The value should be a multiple of 8. If not, some video compressor would have problem.

Attribute

R/W

Remarks

This property is only valid if the AVIVideoSizeByStream is set to False. If AVIVideoSizeByStream is set to be True, it's ignored. All the video frame would be stretched to meat this width if the size is different.

BitmapFile

This property let users set the bitmap to be shown when the control is not playing and histogram data.

Type

String

The name of the bitmap file to be loaded

Attribute

R/W

Remarks

If the server is playing with location data or file based data, the bitmap file will not be shown. The file name could start with “http://”. In such case, the control will retrieve the file from remote server and display it after finishing.

BorderColor

This property is to control the border color shown at the edge of the control.

Type

OLE_COLOR

The color could be either predefined color (with the highest byte contains 80 and the other bytes contain the index) or users defined RGB value (with the highest byte contain 0)

Attribute

R/W

Remarks

For applications that want the control to have the same appearance as before, please set this property to Black color.

CurrentEvent

This property is the index of currently chosen event. This index is only effect when in matched event method.

Type

Long

Attribute

R/W

Remarks

Change of this event would change the playing event immediately. Note this property may be changed internally. For example, when users click on the media map on one event, the control will change current event to that one. So this property will be changed. When this happens, the control will fire an event to notify the current event index changed.

CurrentSegment

This property is to set or get the current segment for current location. Change of this property will stop the current playing.

Type

Long

Attribute

R/W

Remarks

CurrentTimeSec

This property is used to retrieve the current time value (in time_t value defined in C) for playback. This value is only valid when the media starts to play.

Type

Long

Attribute

R/W

Remarks

This value could be used to know where the media is currently played. It contains the same value as OnNotifyTime gives application. But if you try to play and pause immediate for a database or file, the OnNotifyTime might not yet fired, but application could still get the current time value using this property.

DatabasePath

This property is to get the current setting of database path for this control. If you want to change the database path, please use SetDatabasePath method.

Type

String

Attribute

R/O

Remarks

DBLocation

This property is to get the current setting of location.

Type

String

Attribute

R/O

Remarks

This property is read-only. If you want to change the current location, please use SetLocation method.

DBHierarchy

Get or set the way the media files are saved under location path. It determines if sub-directory is created to hold the media files. This property is process-wide. It means if you have more than one controls in your system, change the setting in one control will affect all other controls.

Type

Boolean

True means to create sub-directory, this is the default value. False means not.

Attribute

R/W

Remarks

In FAT32 partition, there is a file number limitation problem. Set this property to true will solve this problem because now each directory will not hold too many files. Sub-Directory solution also solve the problem that if there are large amount of files in the location, use explorer to browse that directory will spend “LONG” time (maybe half an hour).

Deblocking

Get or set the de-blocking 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

De-blocking could solve the blocky problem for video stream in low bandwidth, but it would consume more CPU power when decode. Please set this property carefully because it might cause the system to be insensitive for UI request because of CPU busy on decoding.

Deinterlace

Get or set the deinterlacing mode when decode video stream.

Type

Boolean

True means to deinterlace the video image, False means not. Default value is False.

Attribute

R/W

Remarks

DisplayMotionFrame

Set or get the switch to turn on or turn off the display of motion detection triggered frame (red rectangle).

Type

Boolean

True means to show the motion rectangle when triggered, False means to hide the rectangle. The default value is True.

Attribute

R/W

Remarks

The default value is True.

DisplayPeriod

Set or get the video display period. The setting value is the frame number that would be used to count before one frame is shown. For example, if the set value is 2, then the control will show one frame per two frames it receives. This property is useful to lower down the loading of the computer that running with several controls at the same time.

Type

Long

This is the frame numbers to be set. 0 or 1 means to show every frame, greater value means longer period before screen updated.

Attribute

R/W

Remarks

If the network speed is slow, for example if the control gets only one frame per second, this period count will be ignored. In other word, the DisplayPeriod will be disabled automatically in slow speed link.

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.

EventTypes

This property is to set or get the current searching criteria for event match.

Type

Long

This property is the OR result of the member of the enumeration EDBEventType

Attribute

R/W

Remarks

Change of this property will affect the current match event list and index. It will also affect the media map since the matched events are changed.

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.

ForceNonYUV

Should the control show video in DirectDraw mode (if possible) without YUV surface?

Type

Boolean

Set this value to True to force to display video in non-YUV mode. Default value is False.

Attribute

R/W

Remarks

In some card, using YUV to show video would lead to green screen. In such case, set this flag could still use the DirectDraw capability but solve the green screen problem.

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.

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 current video media in the database is jpeg, this property does not affect. Because the quality is determined when server generates the file. There is no reason to re-generate a jpeg with worse or better (impossible) quality in client side.

MaxEventNumber

This property is used to control the max event number the control could return for the matched event list.

Type

Long

The default value is 5000.

Attribute

R/W

Remarks

You have to use this property carefully. If you change this value too large, and the matched events is too many. It may take a lot of time to finish the query. And more important, you may get a memory allocation error since more events take more memories. And if you set this value too small, you may easily get error when asking for matched event list.

MaxIntervalNumber

This property is used to control the max interval number the control could return for the interval list.

Type

Long

The default value is 5000.

Attribute

R/W

Remarks

You have to use this property carefully. If you change this value too large, and the interval is too many. It may take a lot of time to finish the query. And more important, you may get a memory allocation error since more intervals take more memories. And if you set this value too small, you may easily get error when asking for intervals list.

MediaAOnlyColor

This property is to set or get the color for Audio only time interval drawn on media map.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaAVColor

This property is to set or get the color for Audio/Video time interval drawn on media map.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaCurrentEventColor

This property is to set or get the color for current event drawn on media map.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaDIAlertColor

This property is to set or get the color for DI alert event drawn on media map.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaEmptyColor

This property is to set or get the color for no media time frame drawn on media map. No media time frame means you didn’t record any media into the database during this time period.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaMapBackColor

This property is to set or get the color for the background color for media map. It is used when you don’t specify any database path or location.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaMapVisible

This property is to set or get the visibility of the media map.

Type

Boolean

True means to show the map and False means not to show it.

Attribute

R/W

Remarks

When you set the property to be true, you have also to set the parameter for the map by calling SetMediaMapParam. The control will not show the map until you call that method.

MediaMDEventColor

This property is to set or get the color for motion detection alert event drawn on media map.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

MediaVOnlyColor

This property is to set or get the color for Video only time interval drawn on media map.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting.

PlayFileName

This property is to get the current play file name. If the control is not running in file base mode, the return value could be empty string.

Type

EPlaybackMethod

Attribute

R/W

Remarks

Set of this property will not check if the file is valid or not, any error will be notified by OnPlaybackStatus. If the file name contains “http://”, the control will try to download it at once. The progress will be shown on control caption. During download, the control supports only media playback, all other functions, such as: histogram, slider, or segment/time interval, event, are not available yet. Once the download is finished, status will be called back. Application could use this status callback to initialize the UI objects.

PlaybackMethod

This property is to get the current playback method.

Type

EPlaybackMethod

Attribute

R/O

Remarks

If you want to change the playback method, call SetPlaybackMethod method instead.

PlaybackSpeed

This property is to get or set the playback speed for this control.

Type

EPlaybackSpeed

Attribute

R/W

Remarks

When the speed is changed and the control is playing, the change will reflect immediate to the media been playing.

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

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

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.

SliderBackColor

This property is to set or get the color for the background color of the slider window.

Type

OLE_COLOR

Attribute

R/W

Remarks

You could specify system color or pick any RGB value for this color. The control will convert the system color according current setting. The default color is set to the default dialog background color of window. Note this color also affect the text background of the time period text control under the slider.

SliderVisible

This property is to set or get the visibility of the slider supported by this control.

Type

Boolean

True means to show the slider and false means to hide it.

Attribute

R/W

Remarks

When you set the property to be true, you have also to set the parameter for the map by calling SetSliderParam. The control will not show the slider until you call that method.

TitleBarColor

This property is to control the title bar background color shown at top of the control.

Type

OLE_COLOR

The color could be either predefined color (with the highest byte contains 80 and the other bytes contain the index) or users defined RGB value (with the highest byte contain 0)

Attribute

R/W

Remarks

For applications that want the control to have the same appearance as before, please set this property to Black color.

TitleTextColor

This property is to control the title bar text color shown at top of the control.

Type

OLE_COLOR

The color could be either predefined color (with the highest byte contains 80 and the other bytes contain the index) or users defined RGB value (with the highest byte contain 0)

Attribute

R/W

Remarks

None.

UserDateFormat

Set or get the display video data 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.

3.2 Methods

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.

ChooseAVIAudioCompressor

This method is to choose the audio compressor used when converting AVI.

Syntax

```
HRESULT ChooseAVIAudioCompressor (
    String bstrDialogTitle
    Long *plRet
);
```

Return Value

Always S_OK.

Parameters

bstrDialogFile

[in] This is the caption text for the audio compressor picker dialog.

plRe

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

Remarks

After calling this method, the picked value will be applied to the control immediately (This does not apply to the conversion that starts before the settings change). And the settings are saved in the registry if the AVIRegSubKey value is correct (and users have right to access the registry). So next time when users open this dialog again, the settings will keep the same no matter the control is restarted again or not.

Requirements

None.

ChooseAVIVideoCompressor

This method is to choose the video compressor used when converting AVI.

Syntax

```
HRESULT                          String bstrDialogTitle );
ChooseAVIVideoCompressor (       Long *plRet
```

Return Value

Always S_OK.

Parameters

bstrDialogFile

[in] This is the caption text for the video compressor picker dialog.

plRe

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

Remarks

After calling this method, the picked value will be applied to the control immediately (This does not apply to the conversion that starts before the settings change). And the settings are saved in the registry if the AVIRegSubKey value is correct (and users have right to access the registry). So next time when users open this dialog again, the settings will keep the same no matter the control is restarted again or not.

Requirements

None.

CloseDatabase

This method is to close the currently opened database

Syntax

HRESULT CloseDatabase (	**Long** **plRet*	**);**

Return Value

Always S_OK.

Parameters

plRe

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

Remarks

When this function is called, the exporting AVI or playing will be stopped. And the currently opened location will also be closed. But if the control is repairing database or location, the call for this function will be failed. After this function is called successfully, the control is back to the state that no database is specified. To use the control, application must re-set the database and open location.

Requirements

The SetDatabasePath methods should be called before calling this method.

CloseLocation

This method is to close the currently opened location.

Syntax

HRESULT CloseLocation (	**Long** **plRet*	**);**

Return Value

Always S_OK.

Parameters

plRe

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

Remarks

When this function is called, the exporting AVI or playing will be stopped. But if the control is repairing database or location, the call for this function will be failed. After this function is called successfully, the control is back to the state that database is specified, but no location is opened.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

DeleteLocation

Delete the specified location.

Syntax

HRESULT DeleteLocation (	**String** *strLocation*, **Long** **plRet*	**);**

Return Value

Always S_OK.

Parameters

strLocation

[in] This is the location to be erased.

plRet

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

Remarks

If the control is repairing database or location or if the control has currently opened a location, this method won't work. It works only if the control has set correct database path and no location is open.

Requirements

The SetDatabasePath method should be called before calling this method.

GetCurrentEventInfo

This method is to retrieve the information of current event index.

Syntax

HRESULT GetCurrentEventInfo (	**VARIANT ****pvInfo*	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

pvInfo

[out] The array contains the information of current event. The array is in 4X6 dimension.

- pvInfo[0] contains the starting time of the event, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.
- pvInfo[1] contains the ending time of the event, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.
- pvInfo[2] contains the found time of the event, that is, the time contains frame that could be played for this event. The six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.
- pvInfo[3] only the first two elements are used. The first one contains the event type for this event. It is one of the members in the enumeration EDBEventType. The second one contains value that has different meaning for different event. For window alert event, this is the percentage of alert. If it's DI alert, this value is the millisecond the event happens.

plRe

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

Remarks

If there is no item matched for current event types setting the plRet will return 3001. You should check the return code before handling the array.
For C++ user, the array should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

GetCurrentSegmentPeriod

This method retrieves the time period of current segment.

Syntax

```
HRESULT GetCurrentSegmentPeriod ( Variant *pvStart
                                  Variant *pvEnd
                                  Long *plRet );
```

Return Value

Always S_OK.

Parameters

pvStart

[out] contains the starting time of the time segment, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

pvEnd

[out] contains the ending time of the time segment, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

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

Remarks

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

GetLimitedPeriod

This method retrieves the time period of the limited period you set.

Syntax

HRESULT GetLimitedPeriod (	**Variant** **pvStart*	
	Variant **pvEnd*	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

pvStart

[out] contains the starting time of the time period, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

pvEnd

[out] contains the ending time of the time period, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

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

Remarks

Before you set the limited period by calling SetLimitedPeriod, the limited period is the same as current segment period is. Note that this value will be reset each time when current segment changed.

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

GetLocationList

Retrieve the location list in current database

Syntax

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

Return Value

Always S_OK.

Parameters

pvLocations

[out] The buffer that holds the returned data. It's an array of type String. Each element of the array contains the name of the location.

plRet

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

Remarks

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath method should be called before calling this method.

GetMatchEventList

Retrieve the matched event list.

Syntax

HRESULT GetMatchedEventList (	**Variant ****pvList*,	
	**Long ****plRet*	**);**

Return Value

Always S_OK.

Parameters

pvList

[out] The buffer that holds the returned data. It's a three-dimension array (nx4x6 where n is the number of matched events). pvList[n] contains the same data as GetCurrentEventInfo returns.

plRet

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

Remarks

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

GetSegmentList

Retrieve the time segment list of current location.

Syntax

```
HRESULT GetSegmentList (    Variant *pvList,
                            Long *plRet         );
```

Return Value

Always S_OK.

Parameters

pvList

[out] The buffer that holds the returned data. It's a three-dimension array (nx2x6 where n is the number of segments). pvList[n] contains the starting and ending time of each period.

- pvList[n][0] contains the starting time of the time segment, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.
- pvList[n][1] contains the ending time of the time segment, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

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

Remarks

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before

calling this method.

GetSnapshot

Get the current decoded picture frame.

Syntax

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

Return Value

Always S_OK.

Parameters

eFormat

[in] The format of image caller needs. If the video data of the connection is using Jpeg codec (such as 2K servers or 6K servers running in Jpeg mode), the format could be Jpeg. Otherwise, only BMP is acceptable.

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)

plRet

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

Remarks

If the given picture format is incorrect, error will be returned.

GetTimeIntervalList

Retrieve the time interval list. The list depends on what playback method. If in eFullRange mode, the list will contain all the time intervals in current segment. If the method is eSelectedPeriod, this list would be the intervals that within (or part of the interval is within) the limited period.

Syntax

HRESULT GetTimeIntervalList (	**Variant *** *pvList*, **Long *** *plRet*	**);**

Return Value

Always S_OK.

Parameters

pvList

[out] The buffer that holds the returned data. It's a three-dimension array (nx3x6 where n is the number of intervals). pvList[n] contains the starting and ending time of each period.

- pvList[n][0] contains the starting time of the event, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.
- pvList[n][1] contains the ending time of the event, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.
- pvList[n][2] only three elements are valid.
  - The first one is the media type. 1 means this interval contains only Video, 2 means it contains only Audio. And value 3 means both.
  - The second element is the codec type for audio. Value 256 means G7221, 512 means G729A, 0 means nothing.
  - The second element is the codec type for video. Value 1 means Motion JPGE, 2 means MPEG4 short header mode, 4 means MPEG4, 0 means nothing.

plRet

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

Remarks

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

IsThereMediaAtTimePoint

Test if there is any media at the time point you given.

Syntax

```
HRESULT IsThereMediaAtTimePoint (  VARIANT *pvTime,
                                   Long *plRet          );
```

Return Value

Always S_OK.

Parameters

pvTime

[in] It is an array of 6 elements. The array contains the time to test, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

[out] the return code of the function. 0 means there are media at that time, 3001 means on media at that time, others mean failed.

Remarks

For C++ user, the arrays should be free by calling VariantClear to avoid memory leak.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

NextFrame

When the control is in pause mode, step forward with one queuing frame. So if the current frame on screen is frame 50, call this method to set the frame 51 as the current frame. After calling this method, the control is still in pause state.

Syntax

HRESULT NextFrame (	**Long *** *plRet*	**);**

Return Value

Always S_OK.

Parameters

plRet

[out] the return code of the function. 0 means success, others mean there is no more frame in the system.

Requirements

The control must be in pause mode to call this method, if the control is playing the return value will be 0, but nothing happens.

Pause

Pause the playback.

Syntax

HRESULT Pause (	**Long *** *plRet*	**);**

Return Value

Always S_OK.

Parameters

plRet

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

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

Playback

Start to playback the media. This function will create a new thread to play the media and return immediately.

Syntax

```
HRESULT Playback (                Long *plRet                );
```

Return Value

Always S_OK.

Parameters

plRet

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

Remarks

The playback starts from the start of the time segment if the playback method is eFullRange. And it plays from the beginning of the limited period if the playback method is eSelectedPeriod. If the playback method is eMatchedEvent, the control will play from the start of current event.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

PlayFromTime

Playback from the given time point. If there is no media at the given point, error will be returned. You could not call this function under matched event playback method.

Syntax

```
HRESULT PlayFromTime (      Variant *pvTime,
                            Long *plRet         );
```

Return Value

Always S_OK.

Parameters

pvTime

[in, ref] It is an array of 6 elements. The array contains the time to test, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

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

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

ReleaseBeforeClose

To call the internal release before control closed. This function is meant for C# program. For C#, if application call Exit() function to end the program, the control UI will disappear first, and them the Windows Form. With the help of this method, the strange problem will no longer exist. Other language could also call this function.

Syntax

```
HRESULT ReleaseBeforeClose (    Long *plRet                );
```

Return Value

Always S_OK.

Parameters

plRet

[out] the return code of the function. Always 0.

Remarks

RepairDatabase

Repair the database when it could not opened correctly.

Syntax

HRESULT RepairDatabase (	**String** *strDatabase*, **Long** **plRet*	**);**

Return Value

Always S_OK.

Parameters

strDatabase

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

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 database, and once the repair finish, an event OnPlaybackStatus will be callback with eStatusDBRepairFinish status code, and a True or False for lParam. 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 OnPlaybackStatus will be callback with eStatusLocRepairFinish status code, and a True or False for lParam. 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.

Resume

Resume the playing of media. This method is effective only is called after calling Pause.

Syntax

HRESULT Resume (	**Long ****plRet*	**);**

Return Value

Always S_OK.

Parameters

plRet

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

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

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,
                          Long *plRet             );
```

Return Value

Always S_OK.

Parameters

eFormat

[in] The format of image caller needs. The format of image caller needs.

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. The format of image caller needs.

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.

SetBitmapHandle

Set the bitmap to be shown when the control is not connected to any server.

Syntax

HRESULT SetBitmapHandle (	**Long** *lBitmapHandle*,	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

lBitmapHandle

[in] This is the bitmap handle. Its value is a Win32 BITMAP handle. For example, in VB the Image property of Picture box contains a Handle sub property. And that's the BITMAP handle for the image.

plRet

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

SetDatabasePath

Set the database path for this control.

Syntax

HRESULT SetDatabasePath (	**String** *strPath*,	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

strPath

[in] This is the path to set.

plRet

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

Remarks

When you set a new database path, the original database will be closed. So if you set a incorrect path, the control will be back to path not set mode. Change of database will stop the playing of media.

SetEndOfInterval

Set if the current file is end of a time interval. This is used to let application could control the playback in file mode. If this is set (no matter the value), the control will fire OnRequestNextPlayFileName event when the current file is done. The application should control the play and stop by itself.

Syntax

HRESULT SetEventTypes (	**Boolean** *bEndOfInterval*	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

bEndOfInterval

[in] True if the current file is end of interval. False if not.

plRet

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

SetEventTypes

Set the new event matching condition.

Syntax

HRESULT SetEventTypes (	**Long** *lEventType*,	
	Boolean *bOnlyTest*	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

lEventType

[in] This is the new event matching types to be set.

bOnlyTest

[in] True if you only want to test if the new type could match any event or not, false to set the value. Default value is false.

plRet

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

SetLimitedPeriod

This method set new limited time period.

Syntax

HRESULT SetLimitedPeriod (	**Variant** **pvStart*	
	Variant **pvEnd*	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

pvStart

[in] contains the starting time of the time period, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

pvEnd

[in] contains the ending time of the time period, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

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

Remarks

This method will check the period you give to be ensured that the time period is a subset of the period of current time segment.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

SetLocation

Set the new location.

Syntax

HRESULT SetLocation (	**String** *strLocation*,	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

strLocation

[in] This is the new location to be set.

plRet

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

Requirements

The SetDatabasePath method should be called before calling this method.

SetMediaMapParam

Set the parameter for the media map accessory.

Syntax

HRESULT SetMediaMapParam (	**Long** *lParentHwnd*,	
	Long *lX*	
	Long *lY*	
	Long *lW*	
	Long *lH*	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

lParentHwnd

[in] This is the window handle of the parent window that will contain the media map. Usually, this is the same window that contains the MediaPlayback control.

lX

[in] X coordinate where the media map will be put. The coordinate is related to the parent window, not to the screen. VB user should note that the coordinate unit used by controls in VB is 15 times the normal coordinate used in window API. So if you get the Left value of some control and pass it as x coordinate for media map, remember to divide Left with 15.

lY

[in] Y coordinate where the media map will be put. The coordinate is related to the parent window, not to the screen. VB user should note that the coordinate unit used by controls in VB is 15 times the normal coordinate used in window API. So if you get the Top value of some control and pass it as x coordinate for media map, remember to divide Top with 15.

lW

[in] This is he width for the media map.

lH

[in] This is the height for the media map.

plRet

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

SetPlaybackMethod

Set new playback method for this control.

Syntax

HRESULT SetPlaybackMEthod (	**EPlaybackMEthod** *eMethod*,	
	Boolean *bOnlyTest*,	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

eMethod

[in] This is the new method to be set.

bOnlyTest

[in, default(false)] True if you only want to test if the new method ok to be set, false will set the new method..

plRet

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

Remarks

You have to be careful when change playback method. For example, if you change the event matching type, but there is no matching item. Then the playing will be stop. And you have to handle many other UI enables/disables. So for easier solution, check if the changing is safe before really change it.

SetSliderParam

Set the parameter for the slider accessory.

Syntax

```
HRESULT SetMediaMapParam (  Long lParentHwnd,
                            Long lX
                            Long lY
                            Long lW
                            Long *plRet          );
```

Return Value

Always S_OK.

Parameters

lParentHwnd

[in] This is the window handle of the parent window that will contain the slider. Usually, this is the same window that contains the MediaPlayback control.

lX

[in] X coordinate where the slider will be put. The coordinate is related to the parent window, not to the screen. VB user should note that the coordinate unit used by controls in VB is 15 times the normal coordinate used in window API. So if you get the Left value of some control and pass it as x coordinate for media map, remember to divide Left with 15.

lY

[in] Y coordinate where the slider will be put. The coordinate is related to the parent window, not to the screen. VB user should note that the coordinate unit used by controls in VB is 15 times the normal coordinate used in window API. So if you get the Top value of some control and pass it as x coordinate for media map, remember to divide Top with 15.

lW

[in] This is he width for the media map.

plRet

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

Remarks

The height of slider is a constant, it’s about 50 pixels for normal window API used unit.

StartAVIConversion

Start AVI conversion with the current time boundary. The boundary is either the location time boundary of current time segment or the value set by SetLimitedPeriod.

Syntax

```
HRESULT StartAVIConversion (    Long *plRet                 );
```

Return Value

Always S_OK.

Parameters

plRet

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

Remarks

This function will create another thread to convert the media into AVI and the method returns immediately after the thread created. And during conversion, the control will send event to notify application about the start of new time interval. Application must specify the new file name by setting property AVIFileName. If the file name is not specified or the name is not changed according, the AVI file conversion would either failed with error or the output file will be overwritten again and again.

To stop the conversion before it finishes, please call StopAVIConversion.

After conversion start, any call to SetLocation or SetDatabasePath will be failed with error code VS3ERR_DB_IS_EXPORTING_AVI to indicate that during AVI conversion, it is not permitted to change location or database. But if the control is destroyed during conversion, the exporting will be stopped automatically without finish.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

StartAVIConversionBetween

Start AVI conversion for the data within the specified time boundary.

Syntax

HRESULT StartAVIConversionBetween (	**Variant ****pvDateTimeStart,* **Variant ****pvDateTimeEnd,* **Long ****plRet*	**);**

Return Value

Always S_OK.

Parameters

pvDateTimeStart

[in, ref] It is an array of 6 elements. The array contains the time of the beginning time, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

pvDateTimeEnd

[in, ref] It is an array of 6 elements. The array contains the time of the end time, the six elements are long integers for Year, Month, Day, Hour, Minute, Second for this time.

plRet

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

Remarks

This function will create another thread to convert the media into AVI and the method returns immediately after the thread created. And during conversion, the control will send event to notify application about the start of new time interval. Application must specify the new file name by setting property AVIFileName. If the file name is not specified or the name is not changed according, the AVI file conversion would either failed with error or the output file will be overwritten again and again.

To stop the conversion before it finishes, please call StopAVIConversion.

After conversion start, any call to SetLocation or SetDatabasePath will be failed with error code VS3ERR_DB_IS_EXPORTING_AVI to indicate that during AVI conversion, it is not permitted to change location or database. But if the control is destroyed during conversion, the exporting will be stopped automatically without finish.

Requirements

The SetDatabasePath and Setlocation methods should be called before calling this method.

StopAVIConversion

Stop the AVI conversion if any.

Syntax

```
HRESULT StopAVIConversion (    Long *plRet    );
```

Return Value

Always S_OK.

Parameters

plRet

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

Remarks

When the conversion stopped, an event OnAVINotify will be called to the application. Application could change the UI state for AVI related controls after receiving the event.

StopPlayback

Stop the media playback.

Syntax

HRESULT StopPlayback (	**Long** **plRet*	**);**

Return Value

Always S_OK.

Parameters

plRet

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

Remarks

If the control is in pause state when you call this function. The pause state will be cleared. So next time when you call Playback or PlayFromTime, it will play without pausing.

TestDatabasePath

Test if the database path contains any media locations.

Syntax

HRESULT TestDatabasePath (	**String** *strPath*,	
	Long **plRet*	**);**

Return Value

Always S_OK.

Parameters

strPath

[in] This is the path to be tested.

plRet

[out] the return code of the function. 0 means the database contains at least one location, 3001 means no locations there, others mean failed.

Remarks

This function tests the database path to see if the database exists and if it contains any locations. If the database does not exist, error will be returned. If the database exists but does not contain any locations, a success code VS3SUC_NO_MATCH_ITEM will be returned.

TestLocation

Test if the location contains any media data.

Syntax

HRESULT TestLocation (	**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 tests the location under current database to see if the location exists and if it contains any media data. If the location does not exist, error will be returned. If the location exists but does not contain any media data, a success code VS3SUC_NO_MATCH_ITEM will be returned.

Requirements

The SetDatabasePath method should be called before calling this method.

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 VaControl. 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.

OnAVINotify

The control fires this event for the status of AVI conversion is changed.

Syntax

HRESULT OnAVINotify (	**EAVINotifyStatusCode** *eCode,*	
	Long *lParam1,*	**);**
	Long *lParam2*	**);**

Return Value

Please always return S_OK.

Parameters

eCode

[in] This parameter indicates the code for notification.

lParam1

[in] This parameter indicates the code for notification.

lParam2

[in] This parameter indicates the code for notification.

Remarks

The lParam1 and lParam2 values are determined by the eCode, the following table lists the possible value.

Status	lParam1	lParam2
eConversionStopped	The reason that conversion stops, 1 for StopAVIConversion is called. 0 for the conversion complete.	The KB that has been generated in the AVI files.
eConversionError	The error code why conversion stops	The KB that has been generated in the AVI

		files.

OnNotifyTime

The control fires this event whenever the time of the playing change or user selected time of time period contains no media.

Syntax

HRESULT OnNotifyTime (	**ENotifyTimeType** *eCode* **Variant** **pvTime*	**);**

Return Value

Please always return S_OK.

Parameters

eCode

[in, ref] This parameter indicates the code for notification.

pvTime

[in] This parameter contains the time value of notification. For eNoMatchAtTimePeriod, this is an array contains 12 elements, the fore 6 elements are the starting time (Year, Month, Day, Hour, Minute, Second). And the latter 6 elements are the ending time. For other notification code, it always contains 6 elements for the time point to be notified.

OnPlaybackStatus

The control fires this event whenever the control changes its statuses. It contains a status code to distinguish among different status. Please refer to ENotifyStatusCode to different status.

Syntax

HRESULT OnPlaybackStatus (	**ENotifyStatusCode** *eStatus*,	
	long *lParam1,*	
	long *lParm2*	**);**

Return Value

Please always return S_OK.

Parameters

eStatus

[in] This is the status to be updated.

lParam1

[in] The meaning of this parameter depends on the status. The following table lists the combination for different status and lParam1.

LParam2

[in] The meaning of this parameter depends on the status. The following table lists the combination for different status and lParam2.

Remarks

Status	lParam1	lParam2
eStartPlay	0	0
eStopPlay	The reason code why playing stopped	0
eEventIndexChanged	Old index	New index
ePlaybackMethodChanged	Old method	New method
eLimitedPeriodChanged	0	0

For handling the last status, event catcher should call GetLimitedPeriod to get the new period.

OnRequestAVIFileName

The control fires this event whenever the AVI conversion needs a new name as the AVI file name.

Syntax

```
HRESULT OnRequestAVIFileName (  Variant *pvStartTime,
                                Variant *pvEndTime      );
```

Return Value

Please always return S_OK.

Parameters

pvStartTime

[in, ref] This parameter contains the starting time value for the new time interval to be converted. Application could use this value to format the file name or it could use it's own naming rule to generate the new file name.

pvEndTime

[in, ref] This parameter contains the starting time value for the new time interval to be converted. Application could use this value to format the file name or it could use it's own naming rule to generate the new file name.

Remarks

Application must specify the file name through AVIFileName property.

OnRequestNextPlayFileName

The control fires this event if the control is operated in file base mode and the application has called SetEndOfInterval to set the interval type.

Syntax

```
HRESULT OnRequestNextPlayFileName (                    );
```

Return Value

Please always return S_OK.

Parameters

none

Remarks

Application should set the PlayFileName whenever got this event.

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
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.
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.
1110	VS3ERR_DB_LOC_NOT_INIT	Location hasn't be initialed yet. Most operations could be called only after location opened or created. If you call these methods before location opened or created. This error code would be returned.
1113	VS3ERR_DB_SEGIDX_INCORRECT	The time segment is not correct (out of range)
1114	VS3ERR_DB_PARAM_INCORRECT	Database operation parameter incorrect for methods
1115	VS3ERR_DB_IS_EXPORTING_AVI	The control is currently exporting AVI file, change of location or database is not permitted.
1116	VS3ERR_DB_NOT_EXPORTING_AVI	The control is not exporting AVI file. Call to StopAVIConversion has no effect.

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_INITIALED	VS3ERR_DB_DATABASE_INITIALED
1121	VS3ERR_FBDB_NOT_INIT	File base is not initialized yet
1122	VS3ERR_FBDB_REMOTE_NOT_SUPPORT	The properties or method could not be called before remote file is downloaded OK. Please refer to PlayFileName for more information.
1123	VS3ERR_FBDB_NO_DESCRIPTOR_FILE	The file base needs index file, and the index file is not generated yet, in such case, any methods need index file will return this error code.
1151-1199		Database related internal error
1201-1214		Internal component Error
1301-1307		Memory related error
1414	VS3ERR_EXCEED_MAX_NUM	The searched items are more than limited.
1501-1503		File system error
3001		No match item found
3002		No more frame when call next frame