CHAPTER 6

AxClient API Methods

This chapter lists the methods in the AxClient API, provides detailed information about each method. provides examples for using the methods. It includes the following topics:

Methods for Controlling Video Operations

The following sections describe the methods that provide functionality for controlling a video feed.

Table 6-1 API Method Summary

Method Name	Description
mtStartStream, page 6-3	Loads the designated live or archived video stream into the AxClient and starts playing the stream.
mtStreamStarting, page 6-5	Returns the status of the video stream started by the mtStartStream method.
mtStartStreamWait, page 6-7	Pauses the AxClient for a specified number of milliseconds after invoking the mtStartStream method, before invoking another method, or until the mtStartStream command is finished, whichever occurs first.
playForward, page 6-8	Plays the started archive video stream in the forward direction.
playRewind, page 6-10	Plays the started archive video stream in the reverse direction.
stepForward, page 6-11	Moves the loaded archive video stream one frame in the direction of the current playback.
stepRewind, page 6-12	Moves the loaded archive video stream one frame in the opposite direction of the current playback.
pause, page 6-13	Pauses the playback of an archive or pauses a live source on the current frame.
playResume, page 6-14	Starts playing a previously paused stream and continues in the previous direction of play.
stop, page 6-15	Stops streaming the live or archived video.
close, page 6-16	Stops streaming the feed or archive and disconnects the AXclient from the VSMS host. Also flushes and resets source properties.
setPlayrateEx, page 6-17	Specifies the playback rate for a loaded archive.
repeatUTCSegment, page 6-18	Plays a designated segment of an archive repeatedly (loops the segment).
seekToUTCTime, page 6-19	Seeks to the specified time in an archive. Time is stored as seconds since 1970.
showTimestamp, page 6-21	Shows or hides the timestamp overlay when playing a video source.
addToSync, page 6-22	Allows a client playback window to perform the identical operations that other windows are performing.
removeFromSync, page 6-23	Removes a client playback window from sync control.
createSyncId, page 6-24	Creates a string that can be used for client synchronization.

mtStartStream

HRESULT mtStartStream (
String *source*,
int *version*,
bool *isProxy*);

Purpose

Loads the designated live or archived video stream into the AxClient and starts playing the stream.

Arguments

source	The URI of the video feed.
version	The server version (for all 6.X releases the value is 6).
isProxy	Determines whether source is from a proxy or archive. The *isProxy* argument can be one of the following keywords: • **VARIANT_TRUE** if the source is from a proxy. • **VARIANT_FALSE** if the source is from an archive.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStartOfStream, onMtStartStreamDone.

Notes

This method is equivalent to calling both **switchTo()** and **playForward()** in previous versions of the API. Therefore, **playForward()** does not need to be called **after mtStartStream()** is invoked.

This call does not immediately return success or failure: instead, the **mtStartStreamDone** and **onStartofStream** events must be caught. Only once both events are caught can the application be certain a stream is playing. If only the **mtStartStreamDone** event is caught (and no **onStartofStream** fires) then the client finished with the *mtStartStream* call, but the server did not return a stream to play.

Examples

C# Example

```
// Wrap the AxClient with AXImp and include the wrapped ActiveX dll in your project
try {
   this.axc.mtStartStream(
      bwims://1.1.1.1/p_s1_CiscoHDCamera_1, 6, true);
}
catch(Exception ex) {
   this.logger.subLog("Exception in mtStartStream - " + ex.Message);
}
```

JavaScript example

```
/* Embed the AxClient as an object, and then use this syntax where the embedded object ID
is IMC1. */

var axc = document.applets[“IMC1”];
axc.mtStartStream(bwims://1.1.1.1/p_s1_CiscoHDCamera_1, 6, true);
```

Related methods

close, page 6-16
mtStreamStarting, page 6-5
mtStartStreamWait, page 6-7
setOnEndOfStream, page 6-76
setOnMtStartStreamDone, page 6-77
setOnStartOfStream, page 6-84

mtStreamStarting

HRESULT mtStreamStarting (VARIANT_BOOL *_pVal_);

Purpose

Returns the status of the video stream started by the mtStartStream method.

Arguments

pVal	Indicates whether the stream is starting. The *pVal* argument can be one of the following keywords: • **VARIANT_TRUE** if the stream is starting. • **VARIANT_FALSE** if the stream is not starting, such as when the stream was already started or was not started.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStartOfStream.

Notes

This API method is useful in situations where certain functionality must be performed only after a stream has started.This approach requires the client to perform work to return the state information, however, and does not work well when thread blocking occurs. For example, in the case of a C# application using VMR, the threading model employed may not allow this call to function at all.

The more optimal approach is to catch the **mtStartStreamDone** event in the application, which is fired by the client at the same time this Boolean changes state internally (for example, the same information can be caught instead of polled).

Examples

C# Example

```
if (!this.axc.mtStreamStarting()) {
// Use other AxClient APIs to modify the stream.
}
```

JavaScript Example

```
if(!axc.mtStreamStarting()) { //call other Axclient APIs}
else {//error out}
```

Related Methods

mtStartStream, page 6-3

mtStartStreamWait, page 6-7

setOnMtStartStreamDone, page 6-77

setOnMtStartStreamDone, page 6-77
setOnStartOfStream, page 6-84

mtStartStreamWait

HRESULT mtStartStreamWait (ULONG *msec***);**

Purpose

Pauses the AxClient for a specified number of milliseconds after invoking the **mtStartStream** method, before invoking another method, or until the **mtStartStream** command is finished, whichever occurs first.

Arguments

msec	The number of milliseconds to wait before invoking another method.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

You can use the **mtStartStreamWait** and the **mtStreamStarting** methods to determine whether to wait before invoking another method. If the mtstreamStarting indicates that a stream is starting, you can use the **mtStartStreamWait** to cause the system to wait a few seconds before invoking another method.

Examples

C# Example

```
try {
   this.axc.mtStartStream(
   bwims://1.1.1.1/p_s1_CiscoHDCamera_1, 6, true);
} catch(Exception ex) {
   this.logger.subLog("Exception in mtStartStream - " + ex.Message);
}

this.axc.mtStartStreamWait(2000);

// Use other AxClient APIs to modify the stream.
```

JavaScript Example

```
if (axc.mtStreamStarting()) {
   axc.mtStartStreamWait(2000);
}
// Use other AxClient APIs to modify the stream.
```

Related Methods

mtStartStream, page 6-3

mtStreamStarting, page 6-5

setOnStartOfStream, page 6-84

playForward

HRESULT playForward (void);

Purpose

Plays the started archive video stream in the forward direction.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK

Exceptions

None.

Events Fired

OnStateChanged

Notes

This method applies only to archive video streams.

Examples

C# Example

```
try
   {
       if (axc.getState() == "0")
       {
           axc.playForward();
       }
        else
       {
           //already streaming..
       }
   }
    catch (Exception error)
   {
        this.logger.callbackLog("exception" + er
   }
```

JavaScript Example

```
axc.playForward();
```

Related Methods

playRewind, page 6-10
stepForward, page 6-11
stepRewind, page 6-12
pause, page 6-13
playResume, page 6-14

stop, page 6-15
setOnEndOfStream, page 6-76

playRewind

HRESULT playRewind (void);

Purpose

Plays the started archive video stream in the reverse direction.

Note This API method does not work with MPEG2 archives or archives associated with Bosch cameras.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK

Exceptions

None.

Events Fired

onStateChanged

Notes

This method applies only to archive video streams.

Examples

C# Example

```
this.axc.playRewind();
```

JavaScript Example

```
axc.playRewind();
```

Related Methods

playForward, page 6-8
stepForward, page 6-11
stepRewind, page 6-12
pause, page 6-13
playResume, page 6-14
stop, page 6-15

stepForward

HRESULT stepForward (void);

Purpose

Moves the loaded archive video stream one frame in the direction of the current playback.

Note This API method does not work with MPEG2 archives or archives associated with Bosch cameras.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

This method applies only to archive video streams.

The step forward movement is in the same direction as the current play direction. For example, calling the playRewind method followed by the stepForward method moves the stream one frame backward.

Examples

C# Example

```
this.axc.stepForward();
```

JavaScript Example

```
axc.stepForward();
```

Related Methods

playForward, page 6-8
playRewind, page 6-10
stepRewind, page 6-12
pause, page 6-13
playResume, page 6-14
stop, page 6-15

stepRewind

HRESULT stepRewind (void);

Purpose

Moves the loaded archive video stream one frame in the opposite direction of the current playback.

Note This API method does not work with MPEG2 archives or archives associated with Bosch cameras.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

This method applies only to archive video streams.

Operates in the reverse direction of the current play direction. For example, issuing **playRewind()** then **stepRewind()** moves the stream one step forward.

Examples

C# Example

```
this.axc.stepRewind();
```

JavaScript Example

```
axc.stepRewind();
```

Related Methods

playForward, page 6-8
playRewind, page 6-10
stepForward, page 6-11
pause, page 6-13
playResume, page 6-14
stop, page 6-15

pause

HRESULT pause (void);

Purpose

Pauses the playback of an archive or pauses a live source on the current frame.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

The application must reload the stream (for example, call mtStartStream again) if the stream has been paused for more than 15 minutes. After 15 minutes of pause, the stream will no longer be loaded in memory and no subsequent API calls will have any effect on it.

Examples

C# Example

```
this.axc.pause();
```

JavaScript Example

```
axc.pause();
```

Related Methods

playForward, page 6-8
playRewind, page 6-10
stepForward, page 6-11
stepRewind, page 6-12
playResume, page 6-14
stop, page 6-15
close, page 6-16

playResume

HRESULT playResume (void);

Purpose

Starts playing a previously paused stream and continues in the previous direction of play.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

The application must reload the stream (for example, call **mtStartStream** again) if the stream has been paused for more than 15 minutes. After 15 minutes of pause, the stream will no longer be loaded in memory and no subsequent API calls will have any effect on it.

Examples

C# Example

```
this.axc.playResume();
```

JavaScript Example

```
axc.playResume();
```

Related Methods

playForward, page 6-8
playRewind, page 6-10
stepForward, page 6-11
stepRewind, page 6-12
pause, page 6-13
stop, page 6-15

stop

HRESULT stop (void);

Purpose

Stops streaming the live or archived video.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

We do not recommend that you use this method. Use the close() method instead.

The video data buffer is flushed. However, properties are not unloaded, and get methods will still return valid information for the stream that is loaded. Because of the work associated with the buffer, stop is not recommended if the application will resume playback. If playback resumes within 15 minutes, pause is recommended instead.

Examples

C# Example

```
this.axc.stop();
```

JavaScript Example

```
axc.stop();
```

Related Methods

playForward, page 6-8
playRewind, page 6-10
stepForward, page 6-11
stepRewind, page 6-12
pause, page 6-13
playResume, page 6-14

close

HRESULT close (void)

Purpose

Stops streaming the feed or archive and disconnects the AXclient from the VSMS host. Also flushes and resets source properties.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

To reload the stream after calling the close method, you must use the **mtStartStream** method.

Because of the work associated with flushing the video buffer, we do not recommend using this method if the same video stream is to be viewed again in a relatively short time frame; instead, we recommend that you use the **pause** method.

This method does not have to be called if the client intends to call the **mtStartStream** method for another source in the next moment (for example, changing from one source to another). When the next **mtStartStream** method is initiated, the same buffer flush takes place, so placing a close at the end of a viewing session, right before the next session starts (in the same client), duplicates work.

Examples

C# Example

```
this.axc.close();
```

JavaScript Example

```
axc.close();
```

Related Methods

mtStartStream, page 6-3

pause, page 6-13

setPlayrateEx

HRESULT setPlayrateEx (Long *playRate*)**;**

Purpose

Specifies the playback rate for a loaded archive.

Arguments

playRate	The rate at which a loaded archive is to play. Valid values are 0.05, 0.10, 0.25, 0.50, 0.75, 0.80, 1, 2, 4, 8, 16, 32, and 64: • A value of 1 is the normal play rate. • A value less than 1 is a slower play rate. • A value greater than 1 is a faster play rate.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onPlayrateChanged

Notes

This method replaces the **setPlayrate** method, which should no longer be used.

Examples

C# Example

```
this.axc.setPlayRateEx(8);
```

JavaScript Example

```
axc.setPlayRateEx(8);
```

Related Methods

getPlayrateEx, page 6-37
setOnPlayrateChanged, page 6-79

repeatUTCSegment

HRESULT repeatUTCSegment (
DOUBLE *seekTime*,
LONG *startOffset*,
LONG *endOffset*);

Purpose

Plays a designated segment of an archive repeatedly (loops the segment).

Arguments

seekTime	Start time of the segment in UTC format
startOffset	Specifies how many seconds before the time that seekTime designates the loop to start playing.
endOffset	Specifies how many seconds after the time that seekTime designates the loop to stop playing.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

This method seeks the specified seektime in an archive and repeatedly plays the archive segment bounded by *seekTime – startOffset* and *seekTime – endOffset*. Units are in seconds. The *endOffset* argument is a positive number.

Examples

C# Example

```
this.axc.repeatUTCSegment(12345,1,120);
```

JavaScript Example

```
axc.repeatUTCSegment(12345,1,120);
```

Related Methods

seekToUTCTime, page 6-19
seekToPercentage, page 6-20
setOnSeekTimeChanged, page 6-83
setOnStartTimeChanged, page 6-85
setOnStopTimeChanged, page 6-89

seekToUTCTime

HRESULT seekToUTCTime (DOUBLE *time*);

Purpose

Seeks to the specified time in an archive. Time is stored as seconds since 1970.

Arguments

time	Start time of the segment in UTC format, representing the number of seconds since January 1, 1970.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onStateChanged

Notes

Seeking should not be performed before the **onStartOfStream** event has been received. Wait for the **onStartofStream** event to fire before calling this API. This method replaces the **seek()** method, which should no longer be used and may not function as expected.

Examples

C# Example

```
double startTime = 0;
startTime = axc.getUTCStartTime();
axc.seekToUTCTime(startTime);
```

JavaScript Example

```
axc.seekToUTCTime(12345678);
```

Related Methods

repeatUTCSegment, page 6-18
seekToPercentage, page 6-20
getUTCStartTime, page 6-29
setOnSeekTimeChanged, page 6-83
setOnStartTimeChanged, page 6-85
setOnStopTimeChanged, page 6-89

seekToPercentage

HRESULT seekToPercentage (FLOAT *percent*)**;**

Purpose

Seeks to the specified percentage in an archive.

Arguments

percent	Percentage of the total time of the archive. Valid values are in decimal format between 0 and 1.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

onSeekTimeChanged

Notes

Seeking should not be performed before the **onStartOfStream** event has been received. Wait for the **onStartofStream** event to fire before calling this API. This method replaces the **seek()** method, which should no longer be used and may not function as expected.

Examples

C# Example

```
axc.seekToPercentage(0.56);
```

JavaScript Example

```
axc.seekToPercentage(0.56);
```

Related Methods

repeatUTCSegment, page 6-18
seekToUTCTime, page 6-19
getUTCStartTime, page 6-29
setOnSeekTimeChanged, page 6-83
setOnStartTimeChanged, page 6-85
setOnStopTimeChanged, page 6-89

showTimestamp

HRESULT showTimestamp (VARIANT_BOOL *show*)**;**

Purpose

Shows or hides the timestamp overlay when playing a video source.

The timestamp overlay displays the time associated with each frame of the stream. This call can only be made after the stream is playing. Calling showTimestamp before the stream is actually playing will result in an error. This API only applies to MPEG2 sources.

Arguments

show	Controls whether the timestamp displays: • **VARIANT_TRUE**—Timestamp displays. • **VARIANT_FALSE**—Timestamp does not display.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

By default, timestamps are not displayed.

Examples

C# Example

```
// This example uses a UI checkbox to determine if timestamp should be on or off,
// and then executes that choice once the onStartofStream event has fired
// (timestamp cannot be set prior to this point).

protected void axc_OnStartOfStream(
   object sender,
   _IMediaPlayerCtrlEvents_OnStartOfStreamEvent e)
{
   if (this.timestampOption.Checked) {
      axc.showTimestamp(true);
   } else if (!this.timestampOption.Checked) {
      axc.showTimestamp(false);
   }
}
```

JavaScript Example

```
axc.showTimestamp(true);
```

Related Methods

None.

addToSync

HRESULT addToSync (
BSTR *aSyncId*,
SHORT *aCount*);

Purpose

Allows a client playback window to perform the identical operations that other windows are performing.

This is helpful when viewing a number of archives for a given time period and you desire all the archives to shuttle through the archive set in the same manner (for example, every window starts playing at the same seek point, every window pauses at the same time, etc.).

Arguments

aSyncId	A value against which clients register their UI behavior. Any distinct string is allowed.
aCount	A value greater than 0. Providing a value of 0 will tell the client not to sync against the aSyncId, which is the incorrect way to stop sync from occurring. If you want to stop the sync of a given client, call **removeFromSync** instead.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

This API requires user intervention during the execution of the request. For a headless API to perform the clip save, use the **save** method instead.

For VSM 6.3.2 or higher, this method is supported in both JavaScript and C#; for VSM 6.3, this method is supported only in JavaScript.

Examples

C# Example

```
axc.addToSync("12345678",1);
```

JavaScript Example

```
axc.addToSync("12345678",1);
```

Related Methods

removeFromSync, page 6-23

createSyncId, page 6-24

removeFromSync

HRESULT removeFromSync ();

Purpose

Removes a client playback window from sync control.

Sync control performs the identical operations that other windows are performing, which is helpful when viewing a number of archives for a given time period and you desire all the archives to shuttle through the archive set in the same manner (for example, every window starts playing at the same seek point, every window pauses at the same time, etc.).

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

This API requires user intervention during the execution of the request. For a headless API to perform the clip save, use the **save** method instead.

For VSM 6.3.2 or higher, this method is supported in both JavaScript and C#; for VSM 6.3, this method is supported only in JavaScript.

Examples

C# Example

```
axc.removeFromSync();
```

JavaScript Example

```
axc.removeFromSync();
```

Related Methods

addToSync, page 6-22

createSyncId, page 6-24

createSyncId

HRESULT addToSync (BSTR **aSyncId*);

Purpose

Creates a string that can be used for client synchronization.

Arguments

aSyncId	A value against which clients register their UI behavior. Any distinct string is allowed.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

Not recommended for creating a unique string. A GUID generator produces a more unique string and is recommended instead.

Examples

C# Example

```
string aSyncId = axc.createSyncId();
```

JavaScript Example

```
string aSyncId = axc.createSyncId();
```

Related Methods

addToSync, page 6-22
removeFromSync, page 6-23

Methods for Obtaining Information about the AxClient or Video Streams

The following sections describe the methods that provide functionality for obtaining information about the AxClient and about video streams.

Table 6-2 API Method Summary

Method Name	Description
getCiscoHD, page 6-26	Indicates whether the stream comes from a Cisco high definition IP camera.
getVersion, page 6-27	Retrieves the version number of the AxClient.
getUTCSeekTime, page 6-28	Retrieves the seek time in UTC format.
getUTCStartTime, page 6-29	Retrieves the start time of an archive in UTC format.
getUTCStopTime, page 6-30	Retrieves the last frame time of the archive, in UTC format.
getUTCCurrentTime, page 6-31	Retrieves the current time in an archive, in UTC format.
getUTCOriginalStartTime, page 6-32	Retrieves the actual start time for a loop archive, in UTC format.
getState, page 6-33	Retrieves the state of the player.
getContentType, page 6-34	Retrieves the media type of a loaded source.
getCurrentSource, page 6-35	Retrieves the URL of the currently loaded source.
getRecordrateEx, page 6-36	Retrieves the rate at which the archive was recorded. Depending on the media type of the source, this value is the framerate or the bitrate.
getPlayrateEx, page 6-37	Retrieves the rate at which the loaded archive is playing.
getErrorText, page 6-38	Retrieves the text description of the specified error code.
getProfiles, page 6-39	Retrieves a list of the WMV profiles that the loaded video stream supports.
getProfilesSSV, page 6-40	Retrieves an array of strings that represent the WMV profiles that the loaded video stream supports.
getStreamCodecSubtype, page 6-41	Retrieves the subtype of the codec of the loaded stream.
getVideoWidth, page 6-42	Gets the width of the source stream, in pixels.
getVideoHeight, page 6-43	Gets the height of the source stream, in pixels.
getDisplayWidth, page 6-44	Gets the width in pixels at which the source should be displayed.
getDisplayHeight, page 6-45	Gets the height in pixels at which the source should be displayed.
getX, page 6-46	Retrieves the x coordinate of the center of the view rectangle.
getY, page 6-47	Retrieves the y coordinate of the center of the view rectangle.

getCiscoHD

HRESULT getCiscoHD ();

Purpose

Indicates whether the stream comes from a Cisco high definition IP camera.

Arguments

None.

Return Values

HRESULT S_OK/E_FAIL

pVal	Camera type: • 0—Not a Cisco high definition IP camera • 1—Cisco high definition IP camera

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
int HD = axc.getCiacoHD()
```

JavaScript Example

```
var HD = axc.getCiacoHD()
```

Related Methods

None.

getVersion

HRESULT getVersion (BSTR ****version***);**

Purpose

Retrieves the version number of the AxClient.

Arguments

version	Version number in the format ***n*.*n*.*n*.*n***, where *n* is an integer. For example, 6.2.16.1.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The version of imsclient.dll is returned.

Examples

C# Example

```
String myVersion = axc.getVersion();
```

JavaScript Example

```
var myVersion = axc.getVersion();
```

Related Methods

None.

getUTCSeekTime

HRESULT getUTCSeekTime (DOUBLE *ptime*);

Purpose

Retrieves the seek time in UTC format.

Arguments

ptime	Seek time in UTC format.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

This method is not supported for live feeds.

Examples

C# Example

```
double mySeekTime axc.getUTCSeekTime();
```

JavaScript Example

```
var mySeekTime = axc.getUTCSeekTime();
```

Related Methods

getUTCStartTime, page 6-29
getUTCStopTime, page 6-30
getUTCCurrentTime, page 6-31
getUTCOriginalStartTime, page 6-32

getUTCStartTime

HRESULT getUTCStartTime (DOUBLE *_ptime_);

Purpose

Retrieves the start time of an archive in UTC format.

Arguments

ptime	Start time in UTC format.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the stream is paused, the AxClient does not receive an update for the start or stop times of a loop archive until it resumes playing. For looping archives, the start time of which a paused client is aware could be incorrect (until the client begins playing once again and receives the next start time update).

Examples

C# Example

```
double startTime = 0;
startTime = axc.getUTCStartTime();
axc.seekToUTCTime(startTime);
```

JavaScript Example

```
var startTime = axc.getUTCStartTime();
axc.seekToUTCTime(startTime);
```

Related Methods

seekToUTCTime, page 6-19
seekToPercentage, page 6-20
getUTCSeekTime, page 6-28
getUTCStopTime, page 6-30
getUTCCurrentTime, page 6-31
getUTCOriginalStartTime, page 6-32

getUTCStopTime

HRESULT getUTCStopTime (DOUBLE *_ptime_);

Purpose

Retrieves the last frame time of the archive, in UTC format.

Arguments

ptime	The last frame time of the archive, in UTC format.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the stream is paused, the AxClient does not receive an update for the start or stop times of a loop archive until it resumes playing. For looping archives, the start time of which a paused client is aware could be incorrect (until the client begins playing once again and receives the next start time update).

Examples

C# Example

```
double stopTime = 0;
stopTime = axc.getUTCStopTime();
axc.seekToUTCTime(stopTime);
```

JavaScript Example

```
var stopTime = axc.getUTCStopTime();
axc.seekToUTCTime(stopTime);
```

Related Methods

getUTCSeekTime, page 6-28
getUTCStartTime, page 6-29
getUTCCurrentTime, page 6-31
getUTCOriginalStartTime, page 6-32

getUTCCurrentTime

HRESULT getUTCCurrentTime (Double *_ptime_);

Purpose

Retrieves the current time in an archive, in UTC format.

Arguments

ptime	The current time of an archive, in UTC format.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the stream is paused, the AxClient does not receive an update for the start or stop times of a loop archive until it resumes playing. For looping archives, the start time of which a paused client is aware could be incorrect (until the client begins playing once again and receives the next start time update).

Examples

C# Example

```
double currentTime = 0;
currentTime = axc.getUTCCurrentTime();
axc.seekToUTCTime(currentTime);
```

JavaScript Example

```
var currentTime = axc.getUTCCurrentTime();
```

Related Methods

getUTCSeekTime, page 6-28
getUTCStartTime, page 6-29
getUTCStopTime, page 6-30
getUTCOriginalStartTime, page 6-32

getUTCOriginalStartTime

HRESULT getUTCOriginalStartTime (DOUBLE *_ptime_);

Purpose

Retrieves the actual start time for a loop archive, in UTC format.

Arguments

ptime	Original start time of an archive, in UTC format.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the stream is paused, the AxClient does not receive an update for the start or stop times of a loop archive until it resumes playing. For looping archives, the start time of which a paused client is aware could be incorrect (until the client begins playing once again and receives the next start time update).

Examples

C# Example

```
double startTime = axc.getUTCOriginalStartTime();
```

JavaScript Example

```
var startTime = axc.getUTCOriginalStartTime();
```

Related Methods

getUTCSeekTime, page 6-28
getUTCStartTime, page 6-29
getUTCStopTime, page 6-30
getUTCCurrentTime, page 6-31

getState

HRESULT getState (BSTR *_state_);

Purpose

Retrieves the state of the player.

Arguments

state	The current state of a the player: • 0—Paused • 1—Playing forward • 2—Playing reverse • 3—Seeking • 4—Loading • 5—Stopped • 6—First frame received • –1—(Negative 1) Unknown

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

Based on the called control operation, the state of the AXclient changes, When the state changes, the AX client also fires the **onStateChange** event.

Examples

C# Example

```
string playerState = axc.getState();
```

JavaScript Example

```
var playerState = axc.getState();
```

Related Methods

Methods for Controlling Video Operations, page 6-2

getContentType

HRESULT getContent (BSTR *_state_);

Purpose

Retrieves the media type of a loaded source.

Arguments

state	Media type of the source: • MPEG1 • MPEG2 • MPEG4 • JPEG • H264 • audio

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
string playerContent = axc.getContentType();
```

JavaScript Example

```
var playerContent = axc.getContentType();
```

Related Methods

getCurrentSource, page 6-35
getStreamCodecSubtype, page 6-41

getCurrentSource

HRESULT getCurrentSource (BSTR **pSource*)**;**

Purpose

Retrieves the URL of the currently loaded source.

Arguments

pSource	Video source in the format *//host/source*, where: • *host*—Hostname or IP address of the VMSM host on which the audio source resides • *source*—name of the video source

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
string playerSource = axc.getCurrentSource();
```

JavaScript Example

```
var playerSource = axc.getCurrentSource();
```

Related Methods

getContentType, page 6-34

getRecordrateEx

HRESULT getRecorderateEx (DOUBLE **recordRate*)**;**

Purpose

Retrieves the rate at which the archive was recorded. Depending on the media type of the source, this value is the framerate or the bitrate.

Arguments

recordRate	The rate at which the archive was recorded.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The information that this method provided depends on the media type of the source. For JPEG sources, this value is the framerate. For MPEG sources, this value is the bitrate.

This method does not apply to live feeds. Live feeds return –1.

Caution The frame rate returned by this API (when reporting on a JPEG source) may only represent the frame rate at the client, not the true frame rate at the server.

Examples

C# Example

```
double myFrameRate = axc.getRecordrateEx();
```

JavaScript Example

```
var myFrameRate = axc.getRecordrateEx();
```

Related Methods

getPlayrateEx, page 6-37

getPlayrateEx

HRESULT getPlayrateEx (DOUBLE **playRate*)**;**

Purpose

Retrieves the rate at which the loaded archive is playing.

Arguments

playRate	The rate at which the loaded archive is playing. Valid values are 0.05, 0.10, 0.25, 0.50, 0.75, 0.80, 1, 2, 4, 8, 16, 32, and 64: • A value of 1 is the normal play rate. • A value less than 1 is a slower play rate. • A value greater than 1 is a faster play rate.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
double myFrameRate = axc.getRecordrateEx();
```

JavaScript Example

```
var myFrameRate = axc.getRecordrateEx();
```

Related Methods

setPlayrateEx, page 6-17
getRecordrateEx, page 6-36
setOnPlayrateChanged, page 6-79

getErrorText

HRESULT getErrorText (
VARIANT *errorCode*,
BSTR* *errorText*);

Purpose

Retrieves the text description of the specified error code.

Arguments

errorCode	The numerical value of the error code.
errorText	The text description of the error.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
string strError = axc.getErrorText();
```

JavaScript Example

```
var error = axc.getErrorText();
```

Related Methods

None.

getProfiles

HRESULT getProfiles (
BSTR *source*,
VARIANT **profiles*);

Purpose

Retrieves a list of the WMV profiles that the loaded video stream supports.

Arguments

source	Video source from which to obtain the list of WMV profiles.
profiles	List of WMV profiles.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

CBR means constant bitrate and VBR means variable bitrate.

Examples

C# Example

```
string[] profiles = axc.getProfiles("bwims://10.10.100.200/NorthEntrance");
```

JavaScript Example

```
var profiles = axc.getProfiles();
var profileArray = profile.toArray("bwims://10.10.100.200/NorthEntrance");
```

Related Methods

getProfilesSSV, page 6-40

getProfilesSSV

HRESULT getProfilesSSV (
BSTR *source*,
BSTR* *profiles*);

Purpose

Retrieves an array of strings that represent the WMV profiles that the loaded video stream supports.

Arguments

source	Video source from which to obtain the array of strings.
profiles	Array of strings that represent the WMV profiles.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

CBR means constant bitrate and VBR means variable bitrate.

Examples

C# Example

```
string profiles = axc.getProfilesSSV("bwims://10.10.100.200/NorthEntrance");
```

JavaScript Example

```
var profiles = axc.getProfilesSSV();
var profileArray = profile.toArray("bwims://10.10.100.200/NorthEntrance");
```

Related Methods

getProfiles, page 6-39

getStreamCodecSubtype

HRESULT getStreamCodecSubtype (INT **subtype*)**;**

Purpose

Retrieves the subtype of the codec of the loaded stream.

Arguments

subtype	Subtype of the codec: • 0—JPEG • 6—MPEG4 • 11—H.264

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
int streamTypeInt = axc.getStreamCodecSubtype();
```

JavaScript Example

```
var streamType = axc.getStreamCodecSubtype();
```

Related Methods

getContentType, page 6-34

getVideoWidth

HRESULT getVideoWidth (SHORT **width*);

Purpose

Gets the width of the source stream, in pixels.

Arguments

width	Width of the source stream, in pixels.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The actual source width may differ from the width that the source should be displayed.

Examples

C# Example

```
short videoW= axc.getVideoWidth();
```

JavaScript Example

```
var videoW = axc.getVideoWidth();
```

Related Methods

getVideoHeight, page 6-43
getDisplayWidth, page 6-44
getDisplayHeight, page 6-45

getVideoHeight

HRESULT getVideoHeight (SHORT **height*)**;**

Purpose

Gets the height of the source stream, in pixels.

Arguments

height	Height of the source stream, in pixels.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The actual source height may differ from the height that the source should be displayed.

Examples

C# Example

```
short videoH = axc.getVideoHeight();
```

JavaScript Example

```
var videoH = axc.getVideoHeight();
```

Related Methods

getVideoWidth, page 6-42
getDisplayWidth, page 6-44
getDisplayHeight, page 6-45

getDisplayWidth

HRESULT getDisplayWidth (SHORT **width*)**;**

Purpose

Gets the width in pixels at which the source should be displayed.

Arguments

width	Width of the source stream, in pixels.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The actual source width may differ from the width that the source should be displayed.

Examples

C# Example

```
short displayW = axc.getDisplayWidth();
```

JavaScript Example

```
var displayW = axc.getDisplayWidth();
```

Related Methods

getVideoWidth, page 6-42
getVideoHeight, page 6-43
getDisplayHeight, page 6-45

getDisplayHeight

HRESULT getDisplayHeight (SHORT **height*)**;**

Purpose

Gets the height in pixels at which the source should be displayed.

Arguments

height	Height of the source stream, in pixels.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The actual source height may differ from the height that the source should be displayed.

Examples

C# Example

```
short displayH = axc.getDisplayHeight();
```

JavaScript Example

```
var displayH = axc.getDisplayHeight();
```

Related Methods

getVideoWidth, page 6-42
getVideoHeight, page 6-43
getDisplayWidth, page 6-44

getX

HRESULT getX (DOUBLE *x);

Purpose

Retrieves the x coordinate of the center of the view rectangle.

Arguments

x	The x coordinate of the center of the view rectangle.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
double xCoord = axc.getX();
```

JavaScript Example

```
var xCoord = axc.getX();
```

Related Methods

getY, page 6-47

getY

HRESULT getY (DOUBLE *y);

Purpose

Retrieves the y coordinate of the center of the view rectangle.

Arguments

y	The y coordinate of the center of the view rectangle.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
double yCoord = axc.getY();
```

JavaScript Example

```
var yCoord = axc.getY();
```

Related Methods

getX, page 6-46

Methods for Creating Clips and Snapshots

The following sections describe the methods that provide functionality for creating clips and snapshots from archived video.

Table 6-3 API Method Summary

Method Name	Description
saveInPortableFormat, page 6-49	Saves a clip in WMV format. A profile window pops up once this API has been called, and the user must select the desired WMV format for the saved file.
createCiscoVideoArchive, page 6-51	Creates a clip of the video archive file with the .cva filename extension.
snapshot, page 6-53	Saves the current frame of video to the viewing client computer. Opens a Windows dialog box in which users can choose to save in a number of image formats, including BMP, GIF, JPEG, PNG, and TIFF.
getSnapshotDIB, page 6-54	Creates a snapshot of the current frame in standard Windows device-independent bitmap (DIB) format that can be saved to a .bmp file.
getSnapshotWin32DIB, page 6-55	Creates a snapshot of the current frame in standard Windows device-independent bitmap (DIB) format that can be saved to a .bmp file.

saveInPortableFormat

HRESULT saveInPortableFormat (
BSTR *source*,
BSTR *startTime*,
BSTR *stopTime*,
BSTR *destination*,
BSTR *profile*);

Purpose

Saves a clip in WMV format. A profile window pops up once this API has been called, and the user must select the desired WMV format for the saved file.

Arguments

source	VSMS source URL in the form of *//host/source*.
startTime	Time to start saving, in UTC milliseconds.
stopTime	Time to end saving, in UTC milliseconds.
destination	Location where the clip is saved on the local client. An empty string causes the system to prompt for a destination.
profile	Must be blank.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

This API requires user intervention during the execution of the request. For a headless API to perform the clip save, use the **save** method instead.

Examples

C# Example

```
long startTime = (long)Math.Round(axc.getUTCStartTime());
long endingTime = (long)Math.Round(axc.getUTCStopTime());

//convert the getUTCdate seconds to the milliseconds required by this call
long startTimeMS = startTime * 1000;
long clipEndTimeMS = endingTime * 1000;

try {
   axc.saveInPortableFormat(
      "bwims://myServer/myArchive",
      startTimeMS.ToString(),
      clipEndTimeMS.ToString(),
      "c:\temp.wmv",
```

```
      ""
   );
} catch (Exception ex) {
   threadSafeSet2Text(ex.Message);
}
```

JavaScript Example

```
/*
 * Save Clip (wmv)
 * @param  sourceURL  bwims:// url
 * @param  startUTC   start time in UTC milliseconds
 * @param  stopUTC    end time in UTC milliseconds
 * @param  filePath   local client path to save the generated clip. null value will to
prompt user
 */
clipPortable : function(sourceURL, startUTC, stopUTC, filePath) {
   try {
      if (!this.axc.mtStreamStarting()) {
         sourceURL = new String(sourceURL);
         startUTC  = new String(startUTC);
         stopUTC   = new String(stopUTC);
         filePath  = (null == filePath) ? '' : new String(filePath);
         this.axc.saveInPortableFormat(sourceURL, startUTC, stopUTC, filePath, '');
      } else {
         alerts(TEXT['js-stream-not-loaded']);
      }
   }  catch(ex) { this.setError(ex) }
}
```

Related Methods

createCiscoVideoArchive, page 6-51

setOnSaveResponse, page 6-81

createCiscoVideoArchive

HRESULT createCiscoVideoArchive (BSTR *xmlStreamInfo*)**;**

Purpose

Creates a clip of the video archive file with the .cva filename extension.

Arguments

xmlStreamInfo — The parameters are passed as a string using the following XML schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<createCiscoVideoArchive>
   <layoutName>YourLayoutName</layoutName>
   <clipTime>
      <begin>Start UTC Miliseconds</begin>
      <end>End UTC Miliseconds</end>
   </clipTime>
   <grid>
      <row>
         <cell>
            <cameraName>ARCHIVE_NAME</cameraName>
            <uri>bwims://SERVER_NAME/ARCHIVE_NAME</uri>

         </cell>
         <cell>
            <cameraName>ARCHIVE_NAME</cameraName>
            <uri>bwims://SERVER_NAME/ARCHIVE_NAME</uri>
         </cell>
      </row>

     [Note: Additional rows and cells are allowed]
   </grid>
</createCiscoVideoArchive>
```

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The CVA format supports creating clips with multiple video sources, such as a view in VSOM. The CVA clips can be played using Cisco Review Player.

Examples

C# Example

```
axc.createCiscoVideoArchive("
<?xml version="1.0" encoding="UTF-8"?>
<createCiscoVideoArchive>
   <layoutName>1X1</layoutName>
```

```
   <clipTime>
      <begin>1271021709</begin>
      <end>1271022009</end>
   </clipTime>
   <grid>
      <row>
         <cell>
            <cameraName>My2600Archive</cameraName>
            <uri>bwims://MyServer/My2600Archive</uri>

         </cell>
         <cell>
            <cameraName>My4500Archive</cameraName>
            <uri>bwims://MyServer/My4500Archive</uri>
         </cell>
      </row>
  </grid>
</createCiscoVideoArchive>");
```

JavaScript Example

```
axc.createCiscoVideoArchive("
<?xml version="1.0" encoding="UTF-8"?>
<createCiscoVideoArchive>
   <layoutName>1X1</layoutName>
   <clipTime>
      <begin>1271021709</begin>
      <end>1271022009</end>
   </clipTime>
   <grid>
      <row>
         <cell>
            <cameraName>My2600Archive</cameraName>
            <uri>bwims://MyServer/My2600Archive</uri>

         </cell>
         <cell>
            <cameraName>My4500Archive</cameraName>
            <uri>bwims://MyServer/My4500Archive</uri>
         </cell>
      </row>
  </grid>
</createCiscoVideoArchive>");
```

Related Methods

saveInPortableFormat, page 6-49

setOnSaveResponse, page 6-81

snapshot

HRESULT snapshot (void);

Purpose

Saves the current frame of video to the viewing client computer. Opens a Windows dialog box in which users can choose to save in a number of image formats, including BMP, GIF, JPEG, PNG, and TIFF.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

The file types that are supported by this call are BMP, GIF, JPEG, PNG, and TIFF.

Examples

C# Example

You cannot programmatically save the file: the user must enter data in the pop-up window.

JavaScript Example

You cannot programmatically save the file: the user must enter data in the pop-up window.

Related Methods

getSnapshotDIB, page 6-54
getSnapshotWin32DIB, page 6-55

getSnapshotDIB

HRESULT getSnapshotDIB ();

Purpose

Creates a snapshot of the current frame in standard Windows device-independent bitmap (DIB) format that can be saved to a .bmp file.

Arguments

This method has no arguments.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

Follow the example provided here to generate a proper BMP file.

The difference between **getSnapshotDIB** and **getSnapshotWin32DIB** is that **getSnapshotDIB** returns a properly formed bitmap, while **getSnapshotWin32DIB** returns a bit array including the bitmap info header, the raw bitmap data, and extra characters that must be parsed out before the payload from the API can be consumed.

Examples

C# Example

```
object bmpFromArchive = axc.getSnapshotDIB();

//convert the object to a byte[]
BinaryFormatter bf = new BinaryFormatter();
MemoryStream ms = new MemoryStream();
bf.Serialize(ms, bmpFromArchive);
byte[] data = ms.ToArray();
string clientSnapShotLocation =
   "c:\\Client2Snapshot" +
   DateTime.UtcNow.ToString("yyMMddHmmss") +
   ".bmp";
FileStream fileStream = new FileStream(clientSnapShotLocation, FileMode.Create);
fileStream.Write(data, 0, data.Length);
fileStream.Close();
```

JavaScript Example

None.

Related Methods

snapshot, page 6-53

getSnapshotWin32DIB, page 6-55

getSnapshotWin32DIB

HRESULT getSnapshotWin32DIB (VARIANT **pDIB*)**;**

Purpose

Creates a snapshot of the current frame in standard Windows device-independent bitmap (DIB) format that can be saved to a .bmp file.

Caution The first 13 bytes must be trimmed from the object that is returned in order to get a proper bmp.

Arguments

pDIB	Raw video stream information.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

Follow the example provided here to generate a proper BMP file.

The difference between **getSnapshotDIB** and **getSnapshotWin32DIB** is that **getSnapshotDIB** returns a properly formed bitmap, while **getSnapshotWin32DIB** returns a bit array including the bitmap info header, the raw bitmap data, and extra characters that must be parsed out before the payload from the API can be consumed.

Examples

C# Example

```
//get snapshot, return an object
object bmpFromArchive = axc.getSnapshotWin32DIB();

//convert the object to a byte[]
BinaryFormatter bf = new BinaryFormatter();
MemoryStream ms = new MemoryStream();
bf.Serialize(ms, bmpFromArchive);
byte[] data = ms.ToArray();

// The first 13 bytes need to be trimmed off when writing to file (or using in any other
manner).
// The next 14 bytes need to be formatted as a valid BITMAPFILEHEADER.
int j = 13; //this value may need to change based on VMR control output
int size = (data.Length - j);

//fill the 14 byte header
data[j] = 0x42;
```

```
data[j + 1] = 0x4d;
BitConverter.GetBytes(size).CopyTo(data, j + 2);
data[j + 6] = 0;
data[j + 7] = 0;
data[j + 8] = 0;
data[j + 9] = 0;
data[j + 10] = 0x36;
data[j + 11] = 0;
data[j + 12] = 0;
data[j + 13] = 0;

// Now the byte array is correct, from byte 14 forward
string clientSnapShotLocation = "c:\\Client1Snapshot" +
   DateTime.UtcNow.ToString("yyMMddHmmss") +
   ".bmp";
FileStream fileStream = new FileStream(clientSnapShotLocation, FileMode.Create);
fileStream.Write(data, j, data.Length - j);
fileStream.Close();
```

JavaScript Example

None.

Related Methods

snapshot, page 6-53

getSnapshotDIB, page 6-54

Methods for Controlling VMR Display

The following sections describe the methods that provide functionality for controlling a Video Mixing Renderer (VMR) display.

Table 6-4 API Method Summary

Method Name	Description
setAlpha, page 6-58	Sets the transparency level of the VMR filter.
getAlpha, page 6-59	Retrieves the transparency level of the VMR.
setTransparent, page 6-60	Sets the transparent color and updates the alpha bitmap.
getTransparent, page 6-61	Retrieves the transparent color in the stream.
setBaseRectColor, page 6-62	Sets the base rectangle color and updates the alpha bitmap of the loaded video.
getBaseRectColor, page 6-63	Retrieves the base rectangle color of the loaded stream.
setZoomRectColor, page 6-64	Sets the zoom rectangle color and updates the alpha bitmap of the loaded stream.
getZoomRectColor, page 6-65	Retrieves the zoom rectangle color.
setTimeStampRect, page 6-66	Sets the time stamp rectangle and updates the alpha bitmap.
setVmrDisplayMode, page 6-67	Sets the VMR display mode.
getVmrDisplayMode, page 6-68	Gets the VMR display mode.
setZoomFactor, page 6-69	Sets the zoom factor and updates the display.
getZoomFactor, page 6-70	Retrieves the zoom factor.
move, page 6-71	Changes the size of the viewing rectangle and updates the display.
deltaMove, page 6-72	Changes the view (zoom) rectangle by an increment and updates the display.
resizeVideoWindow, page 6-73	Changes both the rectangle size (zoom) and the video window X and Y co-ordinates.

setAlpha

HRESULT setAlpha (DOUBLE *alpha*)**;**

Purpose

Sets the transparency level of the VMR filter.

Arguments

alpha	Alpha value: • 0—Not transparent • 1—Transparent

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setAlpha(1)
```

JavaScript Example

```
axc.setAlpha(1)
```

Related Methods

getAlpha, page 6-59

getAlpha

HRESULT getAlpha (DOUBLE **alpha*)**;**

Purpose

Retrieves the transparency level of the VMR.

Arguments

None.

Return Values

HRESULT S_OK/E_FAIL

alpha	Alpha value: • 0—Not transparent • 1—Transparent

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
double alpha = axc.getAlpha()
```

JavaScript Example

```
var alpha = axc.getAlpha()
```

Related Methods

setAlpha, page 6-58

setTransparent

HRESULT setTransparent (LONG *rgb*);

Purpose

Sets the transparent color and updates the alpha bitmap.

Arguments

rgb	Microsoft Access color code number that represents the transparent color.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setTransparent(8034025)
```

JavaScript Example

```
axc.setTransparent(8034025)
```

Related Methods

getTransparent, page 6-61

getTransparent

HRESULT getTransparent (LONG **rgb*)**;**

Purpose

Retrieves the transparent color in the stream.

Arguments

rgb	Microsoft Access color code number that represents the transparent color.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
long rgb = axc.getTransparent()
```

JavaScript Example

```
var rgb = axc.getTransparent()
```

Related Methods

setTransparent, page 6-60

setBaseRectColor

HRESULT setBaseRectColor (LONG *rgb***);**

Purpose

Sets the base rectangle color and updates the alpha bitmap of the loaded video.

Arguments

rgb	Microsoft Access color code number that represents the base rectangle color.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setBaseRectColor(13474304)
```

JavaScript Example

```
axc.setBaseRectColor(13474304)
```

Related Methods

getBaseRectColor, page 6-63

getBaseRectColor

HRESULT getBaseRectColor (LONG **rgb****);**

Purpose

Retrieves the base rectangle color of the loaded stream.

Arguments

rgb	Microsoft Access color code number that represents the base rectangle color.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
long rgb = axc.getBaseRectColor()
```

JavaScript Example

```
var rgb = axc.getBaseRectColor()
```

Related Methods

setBaseRectColor, page 6-62

setZoomRectColor

HRESULT setZoomRectColor (LONG *rgb*);

Purpose

Sets the zoom rectangle color and updates the alpha bitmap of the loaded stream.

Arguments

rgb	Microsoft Access color code number that represents the zoom rectangle color.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setZoomRectColor(15643136)
```

JavaScript Example

```
axc.setZoomRectColor(15643136)
```

Related Methods

getZoomRectColor, page 6-65

getZoomRectColor

HRESULT getZoomRectColor (LONG *rgb*)**;**

Purpose

Retrieves the zoom rectangle color.

Arguments

rgb	Microsoft Access color code number that represents the zoom rectangle color.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
long rgb = axc.getZoomRectColor()
```

JavaScript Example

```
var rgb = axc.getZoomRectColor()
```

Related Methods

setZoomRectColor, page 6-64

setTimeStampRect

HRESULT setTimeStampRect (
LONG *left*,
LONG *top*,
LONG *right*,
LONG *bottom*);

Purpose

Sets the time stamp rectangle and updates the alpha bitmap.

Arguments

left	Left coordinate of the rectangle.
top	Top coordinate of the rectangle.
right	Right coordinate of the rectangle.
bottom	Bottom coordinate of the rectangle.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setTimeStampRect(1,3,5,1)
```

JavaScript Example

```
axc.setTimeStampRect(1,3,5,1)
```

Related Methods

None.

setVmrDisplayMode

HRESULT setVmrDisplayMode (SHORT *displayMode*)**;**

Purpose

Sets the VMR display mode.

This places the zoom reticule on screen or hides the zoom reticule (default). This does not enable VMR (the *bag* property associated with imsclient.dll includes this setting and enables VMR).

Arguments

displayMode	VMR display mode: • 0—Do not show the zoom reticule. • 1—Show the zoom reticule. The default value is 0.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setVmrDisplayMode(1);
```

JavaScript Example

```
axc.setVmrDisplayMode(1);
```

Related Methods

getVmrDisplayMode, page 6-68

getVmrDisplayMode

HRESULT getVmrDisplayMode (SHORT **displayMode*)**;**

Purpose

Gets the VMR display mode.

This indicates that either the zoom reticule is on screen or the zoom reticule is hidden (default). This does not enable VMR (the *bag* property associated with imsclient.dll includes this setting and enables VMR).

Arguments

displayMode	VMR display mode: • 0—Does not show the zoom reticule. • 1—Shows the zoom reticule. The default value is 0.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
short displayMode = axc.getVmrDisplayMode();
```

JavaScript Example

```
var displayMode = axc.getVmrDisplayMode();
```

Related Methods

setVmrDisplayMode, page 6-67

setZoomFactor

HRESULT setZoomFactor (DOUBLE *val***);**

Purpose

Sets the zoom factor and updates the display.

Arguments

val	The zoom factor.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
axc.setZoomFactor(2)
```

JavaScript Example

```
axc.setZoomFactor(2)
```

Related Methods

getZoomFactor, page 6-70

getZoomFactor

HRESULT getZoomFactor (DOUBLE **val****);**

Purpose

Retrieves the zoom factor.

Arguments

val	The zoom factor.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

If the VMR is not turned on via the AxClient *bag* property, this call will have no effect.

Examples

C# Example

```
double rgb = axc.getZoomFactor()
```

JavaScript Example

```
var rgb = axc.getZoomFactor()
```

Related Methods

setZoomFactor, page 6-69

move

HRESULT move (
DOUBLE *x*,
DOUBLE *y*);

Purpose

Changes the size of the viewing rectangle and updates the display.

Arguments

x	X coordinate of the viewing rectangle
y	Y coordinate of the viewing rectangle

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

This is the underlying function to which **resizeVideoWindow** forwards.

Examples

C# Example

```
//To view video in a 640x480 rectangle
axc.Move(640, 480);
```

JavaScript Example

```
axc.Move(640,480);
```

Related Methods

deltaMove, page 6-72
resizeVideoWindow, page 6-73

deltaMove

HRESULT deltaMove (
DOUBLE *x*,
DOUBLE *y*);

Purpose

Changes the view (zoom) rectangle by an increment and updates the display.

Arguments

x	X coordinate of the viewing rectangle
y	Y coordinate of the viewing rectangle

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
//To view video in a 640x480 rectangle
axc.deltaMove(640, 480);
```

JavaScript Example

```
axc.deltaMove(640,480);
```

Related Methods

move, page 6-71
resizeVideoWindow, page 6-73

resizeVideoWindow

void resizeVideoWindow (
int *owner*
int *x*,
int *y*,
int *w*,
int *h*);

Purpose

Changes both the rectangle size (zoom) and the video window X and Y co-ordinates.

Arguments

owner	The handle of the target display window
x	New x coordinate of the video window.
y	New y coordinate of the video window.
w	New width of the video window.
h	New height of the video window.

Return Values

HRESULT S_OK/E_FAIL

Exceptions

None.

Events Fired

None.

Notes

This method is similar to the **move** method, except that the **move** method does not actually move the window. It relocates the window to the coordinates provided by the X and Y input parameters. The correct handle of the target display window must be provided to the Owner input parameter or this call will not have a visible impact on the display.

Examples

C# Example

```
//This will resize the window to a 640x480 rectangle
//in the upper left-most corner of the screen
this.axc.ResizeVideoWindow(
   (int)this.axc.Handle,
   1,
   1,
   640,
   480
);
```

JavaScript Example

```
axc.ResizeVideoWindow(
   axc,
   1,
   1,
   640,
   480
);
```

Related Methods

move, page 6-71

deltaMove, page 6-72

Methods for Setting up Callbacks

The following sections describe the methods that provide functionality for setting up callbacks.

Table 6-5 API Method Summary

Method Name	Description
setOnEndOfStream, page 6-76	Invokes the specified user-defined callback function when an archive stops playing.
setOnMtStartStreamDone, page 6-77	Invokes the specified user-defined callback function when an archive playback is initiated.
setOnPlayrateChanged, page 6-79	Invokes the specified user-defined callback function when an archive play rate changes.
setOnSaveResponse, page 6-81	Invokes the specified user-defined callback function when a previously initiated save clip has finished.
setOnSeekTimeChanged, page 6-83	Invokes the specified user-defined callback function when an archive seek time changes.
setOnStartOfStream, page 6-84	Invokes the specified user-defined callback function when an archive playback is initiated.
setOnStartTimeChanged, page 6-85	Invokes the specified user-defined callback function when an archive start time changes.
setOnStateChanged, page 6-87	Invokes the specified user-defined callback function when an archive playback state changes.
setOnStopTimeChanged, page 6-89	Invokes the specified user-defined callback function when an archive stop time changes.

setOnEndOfStream

void setOnEndOfStream (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive stops playing.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when an archive stops playing. user-defined callback function signature: **void callbackFunction (string** *name*); Arguments: • *name*—AxClient object ID.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
axc.setOnEndOfStream('callback_SetOnEndOfStream');
function callback_SetOnEndOfStream(name){}
```

JavaScript Example

```
axc.setOnEndOfStream('callback_SetOnEndOfStream');
function callback_SetOnEndOfStream(name){}
```

Related Methods

mtStartStream, page 6-3
playForward, page 6-8

setOnMtStartStreamDone

void setOnMtStartStreamDone (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive playback is initiated.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when an archive playback is initiated. User-defined callback function signature: **void callbackFunction (string** *name*); Arguments: • *name*—AxClient object ID.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
axc.setOnMtStartStreamDone('callback_SetOnMtStartStreamDone');
function callback_SetOnMtStartStreamDone(name) {

         //Check if the stream is loaded and ready for more commands
         if (axc.mtStreamStarting())
         {
             axc.mtStartStreamWait(100);
         }
}
```

JavaScript Example

```
axc.setOnMtStartStreamDone('callback_SetOnMtStartStreamDone');
function callback_SetOnMtStartStreamDone(name) {

         //Check if the stream is loaded and ready for more commands
         if (axc.mtStreamStarting())
         {
             axc.mtStartStreamWait(100);
         }
}
```

Related Methods

mtStartStream, page 6-3
mtStreamStarting, page 6-5
mtStartStreamWait, page 6-7
setOnStartOfStream, page 6-84

setOnPlayrateChanged

void setOnPlayrateChanged (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive play rate changes.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when an archive play rate changes. User-defined callback function signature: **void callbackFunction (string** *name*, **short** *playRate*); Arguments: • *name*—AxClient object ID. • *playRate*—The rate to which video playback was changed. Valid values are 0.05, 0.10, 0.25, 0.50, 0.75, 0.80, 1, 2, 4, 8, 16, 32, and 64: – A value of 1 is the normal play rate. – A value less than 1 is a slower play rate. – A value greater than 1 is a faster play rate.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
axc.setOnPlayrateChanged('callback_SetOnPlayrateChanged');
function callback_SetOnPlayrateChanged(name, playRate) {}
```

JavaScript Example

```
axc.setOnPlayrateChanged('callback_SetOnPlayrateChanged');
function callback_SetOnPlayrateChanged(name, playRate) {}
```

Related Methods

setPlayrateEx, page 6-17
getPlayrateEx, page 6-37

setOnSaveResponse

void setOnSaveResponse (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when a previously initiated save clip has finished.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when a previously initiated save clip has finished. User-defined callback function signature: **void callbackFunction (** **string** *name*, **bool** *success*, **bool** *confirm*, **string** *location*, **string** *message*); Arguments: • *name*—AxClient object ID. • *success*—Indicates whether the clip was saved successfully. • *confirm*—Indicates whether the clip confirmation was received. • *location*—Location where the clip is saved. Valid values can be one of the following keywords: – **remote**—Saves the clip to the remote VSMS host. – **local**—Saves the clip to the local VSMS host. – **localandremote**—Saves the clip to both the local and remote VSMS hosts. – **user**—Saves the clip to the local PC. • *message*—String. Representation of message returned by VSMS.

Return Values

`None.`

Exceptions

None.

Events Fired

None.

Notes

The server response is the result of the initial save methods.

Examples

C# Example

```
axc.setOnSaveResponse('callback_SetOnSaveResponse');
function callback_SetOnSaveResponse(name, success, confirm, location, message) {}
```

JavaScript Example

```
axc.setOnSaveResponse('callback_SetOnSaveResponse');
function callback_SetOnSaveResponse(name, success, confirm, location, message) {}
```

Related Methods

saveInPortableFormat, page 6-49

createCiscoVideoArchive, page 6-51

setOnSeekTimeChanged

void setOnSeekTimeChanged (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive seek time changes.

Arguments

callbackFunction	The name of the user-defined callback function that is invoked when an archive seek time changes. User-defined callback function signature: **void callbackFunction (** **string** *name*, **Date** *seekTime*); Arguments: • *name*—AxClient object ID. • *seekTime*—Date and time (in UTC format) of the archive after the performed seek operation.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

The callback function is invoked when a seek method has completed.

Examples

C# Example

```
axc.setOnSeekTimeChanged('callback_SetOnSeekTimeChanged');
function callback_SetOnSeekTimeChanged(name, seekTime) {}
```

JavaScript Example

```
axc.setOnSeekTimeChanged('callback_SetOnSeekTimeChanged');
function callback_SetOnSeekTimeChanged(name, seekTime) {}
```

Related Methods

repeatUTCSegment, page 6-18
seekToUTCTime, page 6-19
seekToPercentage, page 6-20

setOnStartOfStream

void setOnStartOfStream (String *callbackFunction*)**;**

Purpose

Invokes the specified user-defined callback function when an archive playback is initiated.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when an archive playback is initiated. User-defined callback function signature: **void callbackFunction (string** *name*)**;** Arguments: • *name*—AxClient object ID.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
axc.setOnStartOfStream('callback_SetOnStartOfStream');
function callback_SetOnStartOfStream(name) {}
```

JavaScript Example

```
axc.setOnStartOfStream('callback_SetOnStartOfStream');
function callback_SetOnStartOfStream(name) {}
```

Related Methods

mtStartStream, page 6-3
mtStreamStarting, page 6-5
mtStartStreamWait, page 6-7
setOnMtStartStreamDone, page 6-77

setOnStartTimeChanged

void setOnStartTimeChanged (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive start time changes.

Arguments

callbackFunction	Name of the user-defined callback function that is invoked when an archive start time changes. User-defined callback function signature: **void callbackFunction (** **string** *name*, **Date** *startTime*); Arguments: • *name*—AxClient object ID. • *startTime*—New start time (in UTC format) of the archive.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

This callback occurs approximately every second when VSMS updates the archive properties or when a new archive source is loaded. When the stream is paused, information is not being passed to AxClient so the start and stop time updates will not trigger.

Examples

C# Example

```
axc.setOnStartTimeChanged('callback_SetOnStartTimeChanged');
function callback_SetOnStartTimeChanged(name, startTime) {}
```

JavaScript Example

```
axc.setOnStartTimeChanged('callback_SetOnStartTimeChanged');
function callback_SetOnStartTimeChanged(name, startTime) {}
```

Related Methods

repeatUTCSegment, page 6-18
seekToUTCTime, page 6-19
seekToPercentage, page 6-20

setOnStateChanged

void setOnOnStateChanged (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive playback state changes.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when an archive playback state changes. User-defined callback function signature: **void callbackFunction (string** *name*, **int** *state*); Arguments: • *name*—AxClient object ID. • *state*—State of the archive playback. Valid values can be one of the following integers: – 0— Paused – 1—Playing Forward – 2—Playing Reverse – 3—Searching/Seeking – 4—Loading – 5—Stopped – 6—First Frame Received

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
axc.setOnStateChanged('callback_SetOnStateChanged');
function callback_SetOnStateChanged(name, state) {}
```

JavaScript Example

```
axc.setOnStateChanged('callback_SetOnStateChanged');
function callback_SetOnStateChanged(name, state) {}
```

Related Methods

Methods for Controlling Video Operations, page 6-2

setOnStopTimeChanged

void setOnStopTimeChanged (String *callbackFunction*);

Purpose

Invokes the specified user-defined callback function when an archive stop time changes.

Arguments

callbackFunction	Name of a user-defined callback function that is invoked when an archive stop time changes. User-defined callback function signature: **void callbackFunction** (**string** *name*, **Date** *stopTime*); Arguments: • *name*—AxClient object ID. • stopTime—New stop time (in UTC format) of the archive.

Return Values

None.

Exceptions

None.

Events Fired

None.

Notes

None.

Examples

C# Example

```
axc.setOnStopTimeChanged('callback_SetOnStopTimeChanged');
function callback_SetOnStopTimeChanged(name, stopTime) {}
```

JavaScript Example

```
axc.setOnStopTimeChanged('callback_SetOnStopTimeChanged');
function callback_SetOnStopTimeChanged(name, stopTime) {}
```

Related Methods

repeatUTCSegment, page 6-18
seekToUTCTime, page 6-19
seekToPercentage, page 6-20