DTPreviewEngine Programmer's Interface Specification

Table of Contents

Overview....4
INSTALLATION....5
Manual Installation....5
Windows....5
OS-X....5
Linux 64 bit....5
CONSTANTS....6
For Play Mode Commands....6
For Time Code Returns....6
For Time Code Source Returns....7
Open Flags....7
COMMANDS....9
Open....9
Close....9
PlayModeSupported....9
PlayMode....10
SetPlayMode....10
SetReadMode....10
Refresh....10
Play....11
PlayAtSpeed....11
PlayFromTo....11
FastForward....11
FastRewind....12
Pause....12
Seek....12
SeekRelative....12
SetTrack....12
Stop....13
Eject....13
Duration....13
LastFrame....13
CurState....14
CurSpeed....14
CurPosition....14
CurTrack....14
CurTcType....15
CurTC....15
CurUB....15
CurDATA....16
Tracks....16
VideoTBCSetup - Deprecated....16
VideoTBCVideo - Deprecated....16
VideoTBCHue - Deprecated....17

VideoTBCChroma - Deprecated....17
VideoTBCGamma – Deprecated....17
AudioOutputLevel....18
AudioBassLevel - Deprecated....18
AudioTrebleLevel - Deprecated....18
SourceMetaDataDWORD....19
SourceMetaDataSTR....26
GetCurExtendedData....30
TargetRect - Deprecated....30
TargetTop - Deprecated....30
TargetLeft - Deprecated....31
TargetWidth - Deprecated....31
TargetHeight - Deprecated....31
TargetAspectRatio....31
GetIn....32
SetIn....32
GetOut....32
SetOut....33
VideoLUT - Deprecated....33
SaveCurFrame....33
SourceFileName....34
SourceHeight....34
SourceWidth....34
SourceBitDepth....34
SourceFourCC....35
SourceRate....35
SourceScale....35
SourceBitRate....36
SourceFrameSize....36
SourceVideoChannels....36
SourceAudioChannels....37
SourceAudioFrequency....37
SourceAudioBitsPerSample....37
SourceAudioFourCC....37
Audio Levels....37
Free....38
SetMode....38
Closed Captions and Safe Zone Overlays....39

Overview

The DrasticPreview Engine allows the caller to play all the Drastic supported audio and video file formats to the VGA, via OpenGL, and a multimedia audio card. It also allows playout, with the correct end user license, to SDI/HDMI using Aja, BlueFish444 or BlackMagic Decklink cards. It uses the same engine as the DrasticPreview, DrasticPreview Pro and the videoQC products (http://www.drasticpreview.com, http://www.videoqc.com) which is based on Drastic's MediaReactor file translation software and DrasticDDR digital disk recorder technology. The API is available as a direct link in Windows, OS-X and Linux 64 (Ubuntu/Debian and Red Hat/Fedora/Centos). This document describes the programmer's API for the DrasticPreview Engine.

INSTALLATION

Installing the DrasicPreview Engine API consist of two parts:

1. Install DrasticPreview, DrasticPreview Pro or videoQC
2. Install the DrasticPreviewAPI sample zip

The DrasticPreview API is an interface DLL included, and used by, DrasticPreview, DrasticPreview Pro and videoQC. It provides the engine, and the samples zip provide the necessary header and library files for linking to it.

Manual Installation

Normally a manual installation is not required, but this information will also help you create your own installers for your products.

Windows

Default installation directory:

C:\Program Files\Drastic\DrasticPreview\

(64 bit, or 32 on 32 bit system)

C:\Program Files (x86)\Drastic\DrasticPreview\

(32 bit)

(**ActiveX Deprecated**) *In most cases all the files in this directory will be required, along with the interface DLL DTPreviewEngine.dll. If you are using the direct link method, this directory will need to be in path so the engine can find its DLLs. If you are using the ActiveX method, you will need to register the DTPreviewEngine.dll as follows:*

Regsvr32 DTPreviewEngine.dll

OS-X

In OS-X, all the required dylib files are kept in the DrasticDDR.framework under /Library/Frameworks. Please note: this folder can be moved to the <user>/Library/Frameworks, or into the <program>.app framework area, but you will have to use the install_name_tool to update the dylib locations.

Linux 64 bit

In Linux 64 bit, the shared objects are stored in /usr/local/lib. To link to the DTPreviewEngine API after installation, you may have to update the library search with: sudo ldconfig

CONSTANTS

For Play Mode Commands

Name	Hex	Decimal
#define PEPLAYMODE_NORMAL	0x0001	// 1
#define PEPLAYMODE_LOOP	0x0002	// 2
#define PEPLAYMODE_PALINDROME	0x0004	// 4
#define PEPLAYMODE_RAM	0x0010	// 16

For Time Code Returns

```
//! Film 24 FPS time code
#define TC2_TCTYPE_FILM          0x00000001   // 1 - 24 fps
//! Non Drop Frame 30 FPS time code
#define TC2_TCTYPE_NDF           0x00000002   // 2 - NTSC Non Drop Frame
//! Drop Frame 29.97 FPS time code
#define TC2_TCTYPE_DF            0x00000004   // 4 - NTSC Drop Frame
//! PAL 25 FPS time code
#define TC2_TCTYPE_PAL           0x00000008   // 8 - PAL
//! Double PAL 50 FPS
#define TC2_TCTYPE_50            0x00000010   // 16 - PAL 720p (double rate)
//! 720p DROP 59.94 FPS
#define TC2_TCTYPE_5994          0x00000020   // 32 - NTSC 59.94fps 720p
//! 720p 60 FPS
#define TC2_TCTYPE_60            0x00000040   // 64 - NTSC 60fps 720p
//! 23.98 FILM for NTSC 23.98 FPS
#define TC2_TCTYPE_NTSCFILM 0x00000080   // 128 - NTSC FILM 23.98
//! Hundredths of a second HH:MM:SS:/100  100 FPS effective
#define TC2_TCTYPE_100           0x00000044   // 68 –
Hours:Minutes:Seconds:Hundreds
```

#define TC2_TCTYPE_IRIG 0x00000045 // Hours:Minutes:Seconds:Xxx

For Time Code Source Returns

//! Using absolute position

#define GS_TCSOURCE_ABS 0

//! Using LTC

#define GS_TCSOURCE_LTC 1

//! Using VITC

#define GS_TCSOURCE_VITC 2

Open Flags

#define VIDUTIL_QUALITY_OPENGL 0x0700 // Use OpenGL Only

#define VIDUTIL_QUALITY_ONLINE 0x0500 // Normal size, hq decomp

#define VIDUTIL_QUALITY_NORMAL 0x0400 // Normal size, normal decomp (default)

#define VIDUTIL_QUALITY_FAST 0x0300 // Fast decomp

#define VIDUTIL_QUALITY_HALF 0x0200 // Full size, One field (a la mtx)

#define VIDUTIL_QUALITY_PREVIEW 0x0100 // Smaller size, Fast decomp

#endif

#ifndef AVHAL_FLAG_FORCE_VGA

//! Force the use of VGA for video

#define AVHAL_FLAG_FORCE_VGA 0x40000000 // Do not use directly

//! Force VGA to an OpenGL buffer

#define AVHAL_FLAG_FORCE_VGA_OPENGL (AVHAL_FLAG_FORCE_VGA | VIDUTIL_QUALITY_OPENGL)

//! Best quality VGA display

#define AVHAL_FLAG_FORCE_VGA_ONLINE (AVHAL_FLAG_FORCE_VGA | VIDUTIL_QUALITY_ONLINE)

```
//! Normal quality VGA display

#define AVHAL_FLAG_FORCE_VGA_NORMAL
(AVHAL_FLAG_FORCE_VGA | VIDUTIL_QUALITY_NORMAL)

//! High frame rate VGA display

#define AVHAL_FLAG_FORCE_VGA_FAST
(AVHAL_FLAG_FORCE_VGA | VIDUTIL_QUALITY_FAST)

//! Half video size VGA display

#define AVHAL_FLAG_FORCE_VGA_HALF
(AVHAL_FLAG_FORCE_VGA | VIDUTIL_QUALITY_HALF)

//! Force VGA display for preview

#define AVHAL_FLAG_FORCE_VGA_PREVIEW
(AVHAL_FLAG_FORCE_VGA | VIDUTIL_QUALITY_PREVIEW)
```

COMMANDS

A = ActiveX - Deprecated

D = DirectX

Open

A: *HRESULT Open([in]BSTR strFileName,[in]long dwFlags,[out,retval]long * plRtn);*
D: DWORD __stdcall DTPreview_Open(const wchar_t * szFileName, void * pHandle, long lPlayMode, long dwFlags, void ** ppPE);

- strFileName – System or UNC path the accessible media file to preview
- pHandle – Pointer to a handle
- lPlayMode – Starting mode for playback
    - #define PEPLAYMODE_NORMAL 0x0001 // 1
    - #define PEPLAYMODE_LOOP 0x0002 // 2
    - #define PEPLAYMODE_PALINDROME 0x0004 // 4
    - #define PEPLAYMODE_RAM 0x0010 // 16
- dwFlags – Flags for the open. Most importantly for OpenGL preview opens (where the caller controls the video window) AVHAL_FLAG_FORCE_VGA_OPENGL
- ppPE – Pointer to a PreviewEngine pointer

Open a new file source for preview.

Close

A: *HRESULT Close([out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_Close(void * pPE);

- pPE – Pointer to preview engine instance

Close the currently open stream or file.

PlayModeSupported

A: *HRESULT PlayModeSupported(long dwTestPlayMode, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_PlayModeSupported(void * pPE, long dwTestPlayMode);

- pPE – Pointer to preview engine instance
- dwTestPlayMode – The play mode you want to test: PEPLAYMODE_NORMAL, PEPLAYMODE_LOOP, PEPLAYMODE_PALINDROME,

PEPLAYMODE_RAM

Returns a bitwise array of supported playback modes (see PlayMode).

PlayMode

A: *HRESULT PlayMode([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_PlayMode(void * pPE);
D: DWORD __stdcall DTPreview_SetPlayMode(void * pPE);

- pPE – Pointer to preview engine instance
- Returns the current play mode: PEPLAYMODE_NORMAL, PEPLAYMODE_LOOP, PEPLAYMODE_PALINDROME, PEPLAYMODE_RAM

Returns the current play mode (1=Normal, 2=Loop, 4=Palindrome, 16=RAM).

SetPlayMode

A: *HRESULT PlayMode([in] long newVal);*
D: DWORD DWORD __stdcall DTPreview_SetPlayMode(void * pPE, long newVal)

- pPE – Pointer to preview engine instance
- newVal – Set the play mode PEPLAYMODE_NORMAL, PEPLAYMODE_LOOP, PEPLAYMODE_PALINDROME, PEPLAYMODE_RAM

Sets the current play mode (1=Normal, 2=Loop, 4=Palindrome, 16=RAM).

SetReadMode

A: *HRESULT PlayMode([in] long newVal);*
D: DWORD DWORD __stdcall DTPreview_SetReadMode(void * pPE, long newVal)

- pPE – Pointer to preview engine instance
- newVal – Set the file read mode to 0 – Force RGB, -1 – Best colour path

Sets the current file read mode (0=Force RGB, -1=Best Case).

Refresh

A: *HRESULT Refresh([out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_Refresh(void * pPE);

- pPE – Pointer to preview engine instance

Refresh the current frame displays in the video window.

Play

A: *HRESULT Play([out,retval] long * lRtn);*
D: DWORD __stdcall DTPreview_Play(void * pPE);

- pPE – Pointer to preview engine instance

Play the current stream at its normal rate.

PlayAtSpeed

A: *HRESULT PlayAtSpeed([in] double lSpeed, [out,retval] long * lRtn);*
D: DWORD __stdcall DTPreview_PlayAtSpeed(void * pPE, double ddSpeed);

- pPE – Pointer to preview engine instance
- ddSpeed – Speed to play at where 1.0 is normal forward play, -1.0 is reverse play, 0.5 is 50% play speed and -2.0 is reverse play twice as fast as normal.

Play the current stream at a specific speed where 1.0 = normal play.

PlayFromTo

A: *HRESULT PlayFromTo([in] long dwFrom, [in] long dwTo, [in] long dwFlags, [in]* double ddSpeed, [out,retval] long * lpRtn);
D: DWORD __stdcall DTPreview_PlayFromTo(void * pPE, long dwFrom, long dwTo, long dwFlags, double ddSpeed);

- pPE – Pointer to preview engine instance
- dwFrom – the first frame to play from
- dwTo – the last frame to play to, which is not played (exclusive)
- dwFlags – play flags (reserved, set to 0)
- ddSpeed – speed to play at (see PlayAtSpeed for more details)

Play the current stream from a start frame to an end frame (end is exclusive).

FastForward

A: *HRESULT FastForward([out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_FastForward(void * pPE);

- pPE – Pointer to preview engine instance

Fast forward the current stream.

FastRewind

A: *HRESULT FastRewind([out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_FastRewind(void * pPE);

- pPE – Pointer to preview engine instance

Fast rewind the current stream.

Pause

A: *HRESULT Pause([out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_Pause(void * pPE);

- pPE – Pointer to preview engine instance

Pause the current stream and display the current frame.

Seek

A: *HRESULT Seek([in] long dwFrame, [out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_Seek(void * pPE, long dwFrame);

- pPE – Pointer to preview engine instance
- dwFrame – the frame to seek to and display

Seek the current stream to a particular frame.

SeekRelative

A: *HRESULT SeekRelative([in] long lOffset, [out,retval] long * lpRtn);*
D: DWORD __stdcall DTPreview_SeekRelative(void * pPE, long lOffset);

- pPE – Pointer to preview engine instance
- lOffset – the positive or negative number of frames to seek from the current position

Seek the current stream to a new position relative to the current position.

SetTrack

A: *HRESULT SetTrack([in] long dwTrack, [out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_SetTrack(void * pPE, long dwTrack);

- pPE – Pointer to preview engine instance
- dwTrack – the selected track for playback

Set the current track in the media or on the device.

Stop

A: *HRESULT Stop([out,retval] long * lpRtn);*
D: DWORD __stdcall DTPreview_Stop(void * pPE);

- pPE – Pointer to preview engine instance

Stop the current stream and return to its beginning (absolute frame 0).

Eject

A: *HRESULT Eject([out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_Eject(void * pPE);

- pPE – Pointer to preview engine instance

Eject the current media or device. Only used for removable media. The 'eject' button in most interfaces is actually a file browser and would not use this command.

Duration

A: *HRESULT Duration([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_Duration(void * pPE);

- pPE – Pointer to preview engine instance

Return the duration (total number of frames) of the media. The frame number of the duration will be one beyond the last physical frame in the file. So if the file is 60 frames long, the available frames will be 0..59 inclusive. To play them all, the in would be 0 and the out would be 60.

LastFrame

A: *HRESULT LastFrame([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_LastFrame(void * pPE);

- pPE – Pointer to preview engine instance

Get last frame that will be displayed. This is normally one frame less than the duration/out point, but can be earlier. This is very useful for play lists, especially where the file length is actually changing during playback.

CurState

A: *HRESULT CurState([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurState(void * pPE);

- pPE – Pointer to preview engine instance

Return the current state of the player (0=stop, 1=pause, 2=play, 5=eject).

CurSpeed

A: *HRESULT CurSpeed([out, retval] double *pVal);*
D: double __stdcall DTPreview_CurSpeed(void * pPE);

- pPE – Pointer to preview engine instance

Return the current speed of the player (1.0=normal play, -1.0=reverse play, 0.0=pause).

CurPosition

A: *HRESULT CurPosition([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurPosition(void * pPE, char * szTC15);

- pPE – Pointer to preview engine instance
- szTC15 – Can be NULL or a pointer to a 15 char (minimum) string to hold the string time code value. e.g. 01:00:00;00

Return the current position (absolute 0..end of stream position). To get the current time code position, please see CurTC below.

CurTrack

A: *HRESULT CurTrack([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurTrack(void * pPE);

- pPE – Pointer to preview engine instance

Return the current track being played (1..Tracks inclusive)

CurTcType

A: *HRESULT CurTcType([in] long dwTcSource, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurTcType(void * pPE, long dwTcSource);

- pPE – Pointer to preview engine instance
- dwTcSource – the time code source. Normal sources include control/absolute time code/frame count, LTC and VITC time code.

Return the current time code type for the LTC or VITC time code. The dwTcSource must be either GS_TCSOURCE_ABS or GS_TCSOURCE_LTC or GS_TCSOURCE_VITC (defined above). It will return TC2_TCTYPE_FILM, TC2_TCTYPE_NDF, TC2_TCTYPE_DF, TC2_TCTYPE_PAL, TC2_TCTYPE_50, TC2_TCTYPE_5994, TC2_TCTYPE_60, TC2_TCTYPE_NTSCFILM or TC2_TCTYPE_100 (also defined above).

CurTC

A: *HRESULT CurTC([in] long dwTcSource, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurTC(void * pPE, long dwTcSource, char * szTC15);

- pPE – Pointer to preview engine instance
- dwTcSource – the time code source. Normal sources include control/absolute time code/frame count, LTC and VITC time code.
- szTC15 – 15 character string for time code string, or NULL

Return the current VITC, LTC or CTL time code. The dwTcSource must be either GS_TCSOURCE_ABS or GS_TCSOURCE_LTC or GS_TCSOURCE_VITC (defined above).

CurUB

A: *HRESULT CurUB([in] long dwTcSource, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurUB(void * pPE, long dwTcSource, char * szUB9);

- pPE – Pointer to preview engine instance
- dwTcSource – the time code source. Normal sources include control/absolute time code/frame count, LTC and VITC time code.
- szUB9 – 9 character string for user bit string, or NULL

Return the current VITC or LTC user bits. The dwTcSource must be either GS_TCSOURCE_LTC or GS_TCSOURCE_VITC (defined above).

CurDATA

A: *HRESULT CurDATA([in] long dwSource, [in] long dwElement, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_CurDATA(void * pPE, long dwSource, long dwElement, long *pData);

- pPE – Pointer to preview engine instance
- dwSource – data source selection
- dwElement – data element selection
- pData – are to place the returned data

Return the current data elements of the frame. This is a custom requester that should only be used if you know the data type available and that it is supported by the SDK. It is NOT normally used.

Tracks

A: *HRESULT Tracks([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_Tracks(void * pPE);

- pPE – Pointer to preview engine instance

Return the total number of tracks available

VideoTBCSetup - Deprecated

A: *HRESULT VideoTBCSetup([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_VideoTBCSetup(void * pPE);

A: *HRESULT VideoTBCSetup([in] long newVal);*
D: DWORD __stdcall DTPreview_SetVideoTBCSetup(void * pPE, long lTBCSetup);

- pPE – Pointer to preview engine instance
- lTBCSetup – setup level for the TBC

Video TBC setup changes the lower black level of the picture (Setup, range 0..65535, -1=unsupported). This only working on compatible display outputs, and may not be supported, depending on hardware.

VideoTBCVideo - Deprecated

A: *HRESULT VideoTBCVideo([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_VideoTBCVideo(void * pPE);

A: *HRESULT VideoTBCVideo([in] long newVal);*

D: DWORD __stdcall DTPreview_SetVideoTBCVideo(void * pPE, long lTBCVideo);

- pPE – Pointer to preview engine instance
- lTBCVideo – video, or white level offset for the TBC

Video TBC Video (Contrast, range 0..65535, -1=unsupported). This only working on compatible display outputs, and may not be supported, depending on hardware.

VideoTBCHue - Deprecated

A: *HRESULT VideoTBCHue([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_VideoTBCHue(void * pPE);

A: *HRESULT VideoTBCHue([in] long newVal);*
D: DWORD __stdcall DTPreview_SetVideoTBCHue(void * pPE, long lTBCHue);

- pPE – Pointer to preview engine instance
- lTBCHue – degree shift for the hue in the TBC

Video TBC Hue (Color shift, range 0..35999 which is equivalent to 0.0..359.99 degrees, -1=unsupported). This only working on compatible display outputs, and may not be supported, depending on hardware.

VideoTBCChroma - Deprecated

A: *HRESULT VideoTBCChroma([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_VideoTBCChroma(void * pPE);

A: *HRESULT VideoTBCChroma([in] long newVal);*
D: DWORD __stdcall DTPreview_SetVideoTBCChroma(void * pPE, long lTBCChroma);

- pPE – Pointer to preview engine instance
- lTBCChroma – chroma level (U/V or Cb/Cr)

Video TBC Chroma (Saturation, range 0..65535, -1=unsupported). This only working on compatible display outputs, and may not be supported, depending on hardware.

VideoTBCGamma – Deprecated

A: *HRESULT VideoTBCGamma([out, retval] double *pVal);*
D: double __stdcall DTPreview_VideoTBCGamma(void * pPE);

A: *HRESULT VideoTBCGamma([in] double newVal);*
D: DWORD __stdcall DTPreview_SetVideoTBCGamma(void * pPE, double

ddGamma);

- pPE – Pointer to preview engine instance
- ddGamma – the gamma curve to set the display to.

Video TBC Gamma correction (-1.0 if not supported). This only working on compatible display outputs, and may not be supported, depending on hardware.

AudioOutputLevel

A: *HRESULT AudioOutputLevel([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_AudioOutputLevel(void * pPE);

A: *HRESULT AudioOutputLevel([in] long newVal);*
D: DWORD __stdcall DTPreview_SetAudioOutputLevel(void * pPE, long lAudioOutputLevel);

- pPE – Pointer to preview engine instance
- lAudioOutputLevel – total audio level, where 0 is silence and 65535 if full volume

Audio Output Level (range 0..65535, -1=unsupported). This may not be supported by all audio outputs.

AudioBassLevel - Deprecated

A: *HRESULT AudioBassLevel([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_AudioBassLevel(void * pPE);

A: *HRESULT AudioBassLevel([in] long newVal);*
D: DWORD __stdcall DTPreview_SetAudioBassLevel(void * pPE, long lAudioBassLevel);

- pPE – Pointer to preview engine instance
- lAudioBassLevel – The amount of bass to add or subtract from the signal. 32768 is nominal (nothing added or subtracted)

Audio Bass Level (range 0..65535, -1=unsupported). This may not be supported by all audio outputs.

AudioTrebleLevel - Deprecated

A: *HRESULT AudioTrebleLevel([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_AudioTrebleLevel(void * pPE);

A: *HRESULT AudioTrebleLevel([in] long newVal);*

D: DWORD __stdcall DTPreview_SetAudioTrebleLevel(void * pPE, long lAudioTrebleLevel);

- pPE – Pointer to preview engine instance
- lAudioTrebleLevel – The amount of bass to add or subtract from the signal. 32768 is nominal (nothing added or subtracted)

Audio Treble Level (range 0..65535, -1=unsupported). This may not be supported by all audio outputs.

SourceMetaDataDWORD

A: *HRESULT SourceMetaDataDWORD([in] long dwMetaDataElement, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceMetaDataDWORD(void * pPE, long dwMetaDataElement, long *plMetaDataDWORD);

- pPE – Pointer to preview engine instance
- dwMetaDataElement – which DWORD based metadata element
- plMetaDataDWORD – returned metadata element

Return source metadata information that are numeric (DWORDs or longs). The currently supported string metadata types are:

```
/** Numeric values for all the metadata information types available in MR and VVW
*/
enum vvwInfoMetaTypes {
        vvwiNumericStart = 0x1000,
        //! see VVWINFO::dwTimeCode
        vvwiTimeCode,
        //! see VVWINFO::dwUserBits
        vvwiUserBits,
        //! see VVWINFO::dwVITCTimeCode
        vvwiVITCTimeCode,
        //! see VVWINFO::dwVITCUserBits
        vvwiVITCUserBits,
        //! see VVWINFO::dwVITCLine3
        vvwiVITCLine3,
        //! see VVWINFO::dwPosterFrame
        vvwiPosterFrame,
        //! see VVWINFO::dwAFrame
        vvwiAFrame,
        //! see VVWINFO::dwAspectRatio
        vvwiAspectRatio,
        //! see VVWINFO::dwOriginalRate
        vvwiOriginalRate,
        //! see VVWINFO::dwOriginalScale
```

```
vvwiOriginalScale,
//! see VVWINFO::dwConversions
vvwiConversions,
//! see VVWINFO::dwVersionNumber
vvwiVersionNumber,
//! see VVWINFO::dwFileSize
vvwiFileSize,
//! see VVWINFO::dwFileDate
vvwiFileDate,
//! see VVWINFO::dwFileTime
vvwiFileTime,
//! see VVWINFO::dwSequenceNumber
vvwiSequenceNumber,
//! see VVWINFO::dwTotalStreams
vvwiTotalStreams,
//! see VVWINFO::dwTotalLength
vvwiTotalLength,
//! see VVWINFO::dwFilmManufacturerCode
vvwiFilmManufacturerCode,
//! see VVWINFO::dwFilmTypeCode
vvwiFilmTypeCode,
//! see VVWINFO::dwWhitePoint
vvwiWhitePoint,
//! see VVWINFO::dwBlackPoint
vvwiBlackPoint,
//! see VVWINFO::dwBlackGain
vvwiBlackGain,
//! see VVWINFO::dwBreakPoint
vvwiBreakPoint,
//! see VVWINFO::dwGamma1000
vvwiGamma1000,
//! see VVWINFO::dwTagNumber
vvwiTagNumber,
//! see VVWINFO::dwFlags
vvwiFlags,
//! see VVWINFO::dwTimeCodeType
vvwiTimeCodeType,
//! see VVWINFO::dwLTCTimeCodeType
vvwiLTCTimeCodeType,
//! see VVWINFO::dwVITCTimeCodeType
vvwiVITCTimeCodeType,
//! see VVWINFO::dwProdDate
vvwiProdDate,
//End: v3.0
//! see VVWINFO::dwUniqueID
vvwiUniqueID,
//!
```

```
        vvwiCustomMetadataBlockType,
        vvwiCustomMetadataBlockSize,
        vvwiNorthSouthEastWest,
        vvwiLatitude,
        vvwiLongitude,
        vvwiExposure,
        vvwiRedGain,
        vvwiBlueGain,

        vvwiEND_OF_DWORD_V2,
        // Add elements here
        //VVVID STRUCT
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoWidth = 0x10000,
        //! XML tag name for width
#define VVWINFOTAG_woVideoWidth                 "Width"
#define VVWINFODESC_woVideoWidth            "Width"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoHeight,
        //! XML tag name for height
#define VVWINFOTAG_woVideoHeight            "Height"
#define VVWINFODESC_woVideoHeight           "Height"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoPlanes,
        //! XML tag name for planes
#define VVWINFOTAG_woVideoPlanes            "Planes"
#define VVWINFODESC_woVideoPlanes           "Planes"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoBitCount,
        //! XML tag name for bit count
#define VVWINFOTAG_woVideoBitCount              "BitCount"
#define VVWINFODESC_woVideoBitCount             "BitCount"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoCompression,
        //! XML tag name for compression (fourcc)
#define VVWINFOTAG_woVideoCompression       "Compression"
#define VVWINFODESC_woVideoCompression      "Compression"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoSizeImage,
        //! XML tag name for size of the image in unsigned chars
#define VVWINFOTAG_woVideoSizeImage             "SizeImage"
```

```
#define VVWINFODESC_woVideoSizeImage            "SizeImage"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoXPelsPerMeter,
        //! XML tag name for X pels per meter
#define VVWINFOTAG_woVideoXPelsPerMeter         "XPelsPerMeter"
#define VVWINFODESC_woVideoXPelsPerMeter        "XPelsPerMeter"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoYPelsPerMeter,
        //! XML tag name for Y pels per meter
#define VVWINFOTAG_woVideoYPelsPerMeter         "YPelsPerMeter"
#define VVWINFODESC_woVideoYPelsPerMeter        "YPelsPerMeter"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoClrUsed,
        //! XML tag name for color elements used
#define VVWINFOTAG_woVideoClrUsed               "ClrUsed"
#define VVWINFODESC_woVideoClrUsed                      "ClrUsed"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoClrImportant,
        //! XML tag name for
#define VVWINFOTAG_woVideoClrImportant          "ClrImportant"
#define VVWINFODESC_woVideoClrImportant         "ClrImportant"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoReserved,
        //! XML tag name for reserved array
#define VVWINFOTAG_woVideoReserved                      "Reserved"
#define VVWINFODESC_woVideoReserved                     "Reserved"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoFccType,
        //! XML tag name for four cc type (video/audio)
#define VVWINFOTAG_woVideoFccType                       "FccType"
#define VVWINFODESC_woVideoFccType                      "FccType"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoFccHandler,
        //! XML tag name for four cc handler
#define VVWINFOTAG_woVideoFccHandler            "FccHandler"
#define VVWINFODESC_woVideoFccHandler           "FccHandler"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoFlags,
        //! XML tag name for flags
```

```
#define VVWINFOTAG_woVideoFlags                 "Flags"
#define VVWINFODESC_woVideoFlags            "Flags"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoCaps,
        //! XML tag name for capabilities
#define VVWINFOTAG_woVideoCaps                  "Caps"
#define VVWINFODESC_woVideoCaps                 "Caps"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoPriority,
        //! XML tag name for priority
#define VVWINFOTAG_woVideoPriority          "Priority"
#define VVWINFODESC_woVideoPriority             "Priority"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoLanguage,
        //! XML tag name for language
#define VVWINFOTAG_woVideoLanguage              "Language"
#define VVWINFODESC_woVideoLanguage             "Language"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoScale,
        //! XML tag name for scale (fps = rate / scale)
#define VVWINFOTAG_woVideoScale                 "Scale"
#define VVWINFODESC_woVideoScale            "Scale"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoRate,
        //!
 XML tag name for rate (fps = rate / scale)
#define VVWINFOTAG_woVideoRate                  "Rate"
#define VVWINFODESC_woVideoRate                 "Rate"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
   vvwiVideoStart,
        //! XML tag name for start frame
#define VVWINFOTAG_woVideoStart                 "Start"
#define VVWINFODESC_woVideoStart            "Start"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoLength,
        //! XML tag name for the length in frames
#define VVWINFOTAG_woVideoLength            "Length"
#define VVWINFODESC_woVideoLength           "Length"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
```

```
        vvwiVideoInitialFrames,
        //! XML tag name for number of initial frames to load
#define VVWINFOTAG_woVideoInitialFrames         "InitialFrames"
#define VVWINFODESC_woVideoInitialFrames   "InitialFrames"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
   vvwiVideoSuggestedBufferSize,
        //! XML tag name for suggested maximum buffer size
#define VVWINFOTAG_woVideoSuggestedBufferSize    "SuggestedBufferSize"
#define VVWINFODESC_woVideoSuggestedBufferSize  "SuggestedBufferSize"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoQuality,
        //! XML tag name for quality
#define VVWINFOTAG_woVideoQuality               "Quality"
#define VVWINFODESC_woVideoQuality                      "Quality"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoSampleSize,
        //! XML tag name for recommended sample size
#define VVWINFOTAG_woVideoSampleSize            "SampleSize"
#define VVWINFODESC_woVideoSampleSize           "SampleSize"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoEditCount,
        //! XML tag name for number of edits done on this file
#define VVWINFOTAG_woVideoEditCount                     "EditCount"
#define VVWINFODESC_woVideoEditCount            "EditCount"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoFormatChangeCount,
        //! XML tag name for number of format changes
#define VVWINFOTAG_woVideoFormatChangeCount             "FormatChangeCount"
#define VVWINFODESC_woVideoFormatChangeCount   "FormatChangeCount"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoPitch,
        //! XML tag name for video line pitch
#define VVWINFOTAG_woVideoPitch                         "Pitch"
#define VVWINFODESC_woVideoPitch                "Pitch"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoDrFlags,
        //! XML tag name for internal drastic flags
#define VVWINFOTAG_woVideoDrFlags               "DrFlags"
#define VVWINFODESC_woVideoDrFlags                      "DrFlags"
        //! INTERNAL: Auto generated for XML output from
```

```
#VVWVIDEO/#VVWAUDIO
        vvwiVideoFileType,
        //! XML tag name for drastic 'mft' file type
#define VVWINFOTAG_woVideoFileType                  "FileType"
#define VVWINFODESC_woVideoFileType                 "FileType"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiVideoResDrastic,
        //! XML tag name for reserved drastic array of DWORDs
#define VVWINFOTAG_woVideoResDrastic            "ResDrastic"
#define VVWINFODESC_woVideoResDrastic           "ResDrastic"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiAudioType,
        //! XML tag
#define VVWINFOTAG_woAudioType                      "AudioType"
#define VVWINFODESC_woAudioType                     "AudioType"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiAudioChannels,
        //! XML tag
#define VVWINFOTAG_woAudioChannels                  "AudioChannels"
#define VVWINFODESC_woAudioChannels                 "AudioChannels"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiAudioFrequency,
        //! XML tag
#define VVWINFOTAG_woAudioFrequency                 "AudioFrequency"
#define VVWINFODESC_woAudioFrequency            "AudioFrequency"
        //! INTERNAL: Auto generated for XML output from
#VVWVIDEO/#VVWAUDIO
        vvwiAudioBits,
        //! XML tag
#define VVWINFOTAG_woAudioBits                      "AudioBits"
#define VVWINFODESC_woAudioBits                     "AudioBits"
        //char  szName[_VVWXXX_NAME_SIZE];      // Stream identifier
        //RECT/*16*/  rcFrame;                         // Frame dimensions
        vvwiLastElementPlus1
        // DO NOT ADD ANYTHING BELOW vvwiLastElementPlus1
};
```

Please see http://www.mediacmd.com for more information.

SourceMetaDataSTR

A: HRESULT SourceMetaDataSTR([in] long dwMetaDataElement, [out, retval] BSTR *pVal);
D: DWORD __stdcall DTPreview_SourceMetaDataSTR(void * pPE, long dwMetaDataElement, wchar_t ** ppWCMetaDataStr);

- pPE – Pointer to preview engine instance
- dwMetaDataElement – which DWORD based metadata element
- ppWCMetaDataStr – returned metadata string

Return source metadata information that are string data. When using the direct api, free the returned memory with DTPreview_Free. The currently supported string metadata types are:

```
/** Numeric values for all the metadata information types available in MR and VVW
*/
enum vvwInfoMetaTypes {
        //! see VVWINFO::szFileName
        vvwiFileName,
        //! see VVWINFO::szNativeLocator
        vvwiNativeLocator,
        //! see VVWINFO::szUniversalName
        vvwiUniversalName,
        //! see VVWINFO::szIP
        vvwiIP,
        //! see VVWINFO::szSourceLocator
        vvwiSourceLocator,

        //! see VVWINFO::szChannel
        vvwiChannel,
        //! see VVWINFO::szChannelName
        vvwiChannelName,
        //! see VVWINFO::szChannelDescription
        vvwiChannelDescription,
        //! see VVWINFO::szTitle
        vvwiTitle,
        //! see VVWINFO::szSubject
        vvwiSubject,
        //! see VVWINFO::szCategory
        vvwiCategory,                           // <-- 10
        //! see VVWINFO::szKeywords
        vvwiKeywords,
        //! see VVWINFO::szRatings
        vvwiRatings,
        //! see VVWINFO::szComments
        vvwiComments,
        //! see VVWINFO::szOwner
```

```
vvwiOwner,
//! see VVWINFO::szEditor
vvwiEditor,
//! see VVWINFO::szSupplier
vvwiSupplier,
//! see VVWINFO::szSource
vvwiSource,
//! see VVWINFO::szProject
vvwiProject,
//! see VVWINFO::szStatus
vvwiStatus,
//! see VVWINFO::szAuthor
vvwiAuthor,                          // <-- 20
//! see VVWINFO::szRevisionNumber
vvwiRevisionNumber,
//! see VVWINFO::szProduced
vvwiProduced,
//! see VVWINFO::szAlbum
vvwiAlbum,
//! see VVWINFO::szArtist
vvwiArtist,
//! see VVWINFO::szComposer
vvwiComposer,
//! see VVWINFO::szCopyright
vvwiCopyright,
//! see VVWINFO::szCreationData
vvwiCreationData,
//! see VVWINFO::szDescription
vvwiDescription,
//! see VVWINFO::szDirector
vvwiDirector,
//! see VVWINFO::szDisclaimer
vvwiDisclaimer,                      // <-- 30
//! see VVWINFO::szEncodedBy
vvwiEncodedBy,
//! see VVWINFO::szFullName
vvwiFullName,
//! see VVWINFO::szGenre
vvwiGenre,
//! see VVWINFO::szHostComputer
vvwiHostComputer,
//! see VVWINFO::szInformation
vvwiInformation,
//! see VVWINFO::szMake
vvwiMake,
//! see VVWINFO::szModel
vvwiModel,
```

```
//! see VVWINFO::szOriginalArtist
vvwiOriginalArtist,
//! see VVWINFO::szOriginalFormat
vvwiOriginalFormat,
//! see VVWINFO::szPerformers
vvwiPerformers,                         // <-- 40
//! see VVWINFO::szProducer
vvwiProducer,
//! see VVWINFO::szProduct
vvwiProduct,
//! see VVWINFO::szSoftware
vvwiSoftware,
//! see VVWINFO::szSpecialPlaybackRequirements
vvwiSpecialPlaybackRequirements,
//! see VVWINFO::szTrack
vvwiTrack,
//! see VVWINFO::szWarning
vvwiWarning,
//! see VVWINFO::szURLLink
vvwiURLLink,
//! see VVWINFO::szEditData1
vvwiEditData1,
//! see VVWINFO::szEditData2
vvwiEditData2,
//! see VVWINFO::szEditData3
vvwiEditData3,                          // <-- 50
//! see VVWINFO::szEditData4
vvwiEditData4,
//! see VVWINFO::szEditData5
vvwiEditData5,
//! see VVWINFO::szEditData6
vvwiEditData6,
//! see VVWINFO::szEditData7
vvwiEditData7,
//! see VVWINFO::szEditData8
vvwiEditData8,
//! see VVWINFO::szEditData9
vvwiEditData9,
//! see VVWINFO::szVersionString
vvwiVersionString,
//! see VVWINFO::szManufacturer
vvwiManufacturer,
//! see VVWINFO::szLanguage
vvwiLanguage,
//! see VVWINFO::szFormat
vvwiFormat,                             // <-- 60
//! see VVWINFO::szInputDevice
```

```
        vvwiInputDevice,
        //! see VVWINFO::szDeviceModelNum
        vvwiDeviceModelNum,
        //! see VVWINFO::szDeviceSerialNum
        vvwiDeviceSerialNum,
        //! see VVWINFO::szReel
        vvwiReel,
        //! see VVWINFO::szShot
        vvwiShot,
        //! see VVWINFO::szTake
        vvwiTake,
        //! see VVWINFO::szSlateInfo
        vvwiSlateInfo,
        //! see VVWINFO::szFrameAttribute
        vvwiFrameAttribute,
        //! see VVWINFO::szEpisode
        vvwiEpisode,
        //! see VVWINFO::szScene
        vvwiScene,                              // <-- 70
        //! see VVWINFO::szDailyRoll
        vvwiDailyRoll,
        //! see VVWINFO::szCamRoll
        vvwiCamRoll,
        //! see VVWINFO::szSoundRoll
        vvwiSoundRoll,
        //! see VVWINFO::szLabRoll
        vvwiLabRoll,
        //! see VVWINFO::szKeyNumberPrefix
        vvwiKeyNumberPrefix,
        //! see VVWINFO::szInkNumberPrefix
        vvwiInkNumberPrefix,
        //! see VVWINFO::szPictureIcon
        vvwiPictureIcon,
        //! see VVWINFO::szProxyFile
        vvwiProxyFile,
        //!
        vvwiCustomMetadataBlockPointer,
        //!
        vvwiImageInfo,
        //!
        vvwiUMID,
        //
        vvwiEND_OF_STRINGS,
};
```

Please see http://www.mediacmd.com for more information.

GetCurExtendedData

A: *HRESULT GetCurExtendedData([in,out] VARIANT *pvData, [in,out] long *plSize);*

D: DWORD __stdcall DTPreview_GetCurExtendedData(void * pPE, BYTE ** ppvData, long * plSize);

- pPE – Pointer to preview engine instance
- ppvData – pointer to extended data
- lpSize – pointer to reveive the size fo the extended data

Get current extended frame data. This is custom frame data, embedded with each video frame. It is completely dependent on the source file and should not be used unless you are VERY familiar with the format of the data. In direct mode, free the returned memory with DTPreview_Free.

TargetRect - Deprecated

A: *HRESULT TargetRect([in] long dwX, [in] long dwY, [in] long dwWidth, [in] long dwHeight, [out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_TargetRect(void * pPE, long dwX, long dwY, long dwWidth, long dwHeight);

- pPE – Pointer to preview engine instance
- dwX – left most point in pixels
- dwY – top most point in pixels
- dwWidth – width in pixels
- dwHeight – height in lines

Set the target rectangle for the video window. On older versions of DrasticPreview, this had to be either full size (of the original video image), ½ or ¼ sized. More recent SDKs support arbitrary sizing of the output video window.
For Visual Basic Users: This call must be made with lines and pixels. To make sure you are using pixels for your form, make sure your ScaleMode is set to "3 – pixels" before calling TargetRect with the .Width/.Height or .ScaleWidth/.ScaleHeight.

TargetTop - Deprecated

A: *HRESULT TargetTop([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_TargetTop(void * pPE);

- pPE – Pointer to preview engine instance

Target frame top position (Y absolute coordinate)

TargetLeft - Deprecated

A: *HRESULT TargetLeft([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_TargetLeft(void * pPE);

- pPE – Pointer to preview engine instance

Target frame left position (X absolute coordinate)

TargetWidth - Deprecated

A: *HRESULT TargetWidth([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_TargetWidth(void * pPE);

- pPE – Pointer to preview engine instance

Target frame width in pixels (CX delta value).

TargetHeight - Deprecated

A: *HRESULT TargetHeight([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_TargetHeight(void * pPE);

- pPE – Pointer to preview engine instance

Target frame height in lines (CY delta value).

TargetAspectRatio

A: *HRESULT TargetAspectRatio([in] double valOne, [in] double valTwo);*

D: DWORD __stdcall DTPreview_SetTargetAspectRatio(void * pPE, double ddAspect1, double ddAspect2);

- pPE – Pointer to preview engine instance
- ddAspect1 – the left aspect ration (the 16 in 16x9 or 4 in 4x3)

- ddAspect2 – the right aspect ration (the 9 in 16x9 or 3 in 4x3)

Set target aspect ratio. This is to set the aspect ratio of your output display. It defaults to 4:3, which is the aspect ratio of most computer monitors, projectors and televisions. Some projectors and monitors have a 16:9 aspect ratio which can be compensated for here.

GetIn

A: *HRESULT In([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_GetIn(void * pPE);

- pPE – Pointer to preview engine instance

Returns the current in point (see SetIn). This is the first frame to be displayed in a preview or loop. It is included and displayed, unlike the out point.

SetIn

A: *HRESULT In([in] long newVal);*

D: DWORD __stdcall DTPreview_SetIn(void * pPE, long lIn);

- pPE – Pointer to preview engine instance
- lIn – the inpoint in frames

Set the in (or start) point. This is the first frame to be displayed in a preview or loop. It is included and displayed, unlike the out point.

GetOut

A: *HRESULT Out([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_GetOut(void * pPE);

- pPE – Pointer to preview engine instance

Returns the current out (or end) point. This is one frame AFTER the last frame to display. The out point is never included, so a file with an outpoint set at frame 60 (2:00 NDF) will display 60 frames from 0..59 inclusive. It will never reach 60.

SetOut

A: *HRESULT Out([in] long newVal);*

D: DWORD __stdcall DTPreview_SetOut(void * pPE, long lOut);

- pPE – Pointer to preview engine instance
- lOut – the outpoint in frames

Set the out (or end) point. This is one frame AFTER the last frame to display. The out point is never included, so a file with an outpoint set at frame 60 (2:00 NDF) will display 60 frames from 0..59 inclusive. It will never reach 60.

VideoLUT - Deprecated

A: *HRESULT VideoLUT([in] long dwLUT, [in] long dwType);*

D: DWORD __stdcall DTPreview_SetVideoLUT(void * pPE, long dwLUT, long dwType);

- pPE – Pointer to preview engine instance

Set video look up table (LUT). This is currently only used for 10 bit linear/logarithmic frame types such as DPX, Cineon and 10 bit per component RGB QuickTime Movie files.

SaveCurFrame

A: *HRESULT SaveCurFrame([in] BSTR bstrFileName, [in] long dwParam, [out,retval] long * plRtn);*
D: DWORD __stdcall DTPreview_SaveCurFrame(void * pPE, const wchar_t * szFileName, long dwFrame);

- pPE – Pointer to preview engine instance
- szFileName – file name and path to save frame to
- dwFrame – the frame number to save

Save the current frame as a still image. This is not currently implemented, but will eventually support JPG, BMP as well as YUV or DPX for some files.

SourceFileName

A: *HRESULT SourceFileName([out, retval] BSTR *pVal);*
D: DWORD __stdcall DTPreview_SourceFileName(void * pPE, wchar_t * pwcSourceFileName);

- pPE – Pointer to preview engine instance
- pwcSourceFileName – the source name as a string

The final file name used for the source file. This may be altered from the file name that was used to open the file. For instance, opening a sequence of stills with V:\Test\NotTest_0002.dpx would return V:\Test\NotTest_*.dpx.

SourceHeight

A: *HRESULT SourceHeight([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceHeight(void * pPE);

- pPE – Pointer to preview engine instance

Source video media's height in lines.

SourceWidth

A: *HRESULT SourceWidth([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceWidth(void * pPE);

- pPE – Pointer to preview engine instance

Source video media's width in pixels.

SourceBitDepth

A: *HRESULT SourceBitDepth([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceBitDepth(void * pPE);

- pPE – Pointer to preview engine instance

Source video media's bit depth. Normally 1, 4, 8, 12, 15, 16, 24, 32, 48 or 64. This is for all components in 1 pixel. For instance, an MPEG file may have 8 bit precision (per component), but would still have 12 (4:2:0) or 16 (4:2:2) bits per pixel (bit depth). In RGB a 24 bit depth TGA image would still have three 8 bit components (R:8, G:8, B:8).

SourceFourCC

A: *HRESULT SourceFourCC([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceFourCC(void * pPE);

- pPE – Pointer to preview engine instance

Source video media's fourcc compression code. FourCC stands for Four Character Code. This is the most basic description of a codec in video for windows, QuickTime and other file formats. Here it is extended to all supported types for compatibility. It is usually four normal characters (human readable), but can also be 0 (RGB), 1..4 (RGB Bit Fields) or other non displayable values.

SourceRate

A: *HRESULT SourceRate([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_SourceRate(void * pPE);

- pPE – Pointer to preview engine instance

Source video frame rate 'rate' value (FPS = SourceRate / SourceScale). Around 30 is generally NTSC, 25 is PAL and 24/23.9 is FILM. HD can use any of these frame rates.

Typical Values:	Rate	Scale	FPS
	30	1	30
	30000	1001	29.97...
	2970	100	29.97
	25	1	25
	24	1	24
	24000	1001	23.98

SourceScale

A: *HRESULT SourceScale([out, retval] long *pVal);*

D: DWORD __stdcall DTPreview_SourceScale(void * pPE);

- pPE – Pointer to preview engine instance

Source video frame rate 'scale' value (FPS = SourceRate / SourceScale). Around 30 is generally NTSC, 25 is PAL and 24/23.9 is FILM. HD can use any of these frame rates.

Typical Values:	Rate	Scale	FPS

30	1	30
30000	1001	29.97...
2970	100	29.97
25	1	25
24	1	24
24000	1001	23.98

SourceBitRate

A: *HRESULT SourceBitRate([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceBitRate(void * pPE);

- pPE – Pointer to preview engine instance

Source video media's bit rate in kilobytes per second. Multiply by 1024 to get bytes per second or 8096 to get bits per second. This is the overall bit rate for any streams in the main file/sequence. It is for reference only and should not be considered canonical.

SourceFrameSize

A: *HRESULT SourceFrameSize([in] long dwFrame, [out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceFrameSize(void * pPE, long dwFrame);

- pPE – Pointer to preview engine instance
- dwFrame – the frame to get the size of

Source video media's frame size in bytes for the requested frame (dwFrame) or current frame (if dwFrame == -1).

SourceVideoChannels

A: *HRESULT SourceVideoChannels([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceVideoChannels(void * pPE);

- pPE – Pointer to preview engine instance

Source video total channels. This is an absolute figure (e.g. 6 for six channels) and not a bit wise figure (which would be hex 3f for six channels).

SourceAudioChannels

A: *HRESULT SourceAudioChannels([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceAudioChannels(void * pPE);

- pPE – Pointer to preview engine instance

Source audio total channels. This is an absolute figure (e.g. 6 for six channels) and not a bit wise figure (which would be hex 3f for six channels).

SourceAudioFrequency

A: *HRESULT SourceAudioFrequency([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceAudioFrequency(void * pPE);

- pPE – Pointer to preview engine instance

Source audio media frequency

SourceAudioBitsPerSample

A: *HRESULT SourceAudioBitsPerSample([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceAudioBitsPerSample(void * pPE);

- pPE – Pointer to preview engine instance

Source audio media bits per sample. Depending on the compression type, this may be completely accurate, rough or inaccurate. Mostly correct for RGB types, but YCbCr types are normally marked as 16 bits per pixel, which is somewhat correct for 8 bits per component, but incorrect for 10 bits per component.

SourceAudioFourCC

A: *HRESULT SourceAudioFourCC([out, retval] long *pVal);*
D: DWORD __stdcall DTPreview_SourceAudioFourCC(void * pPE);

- pPE – Pointer to preview engine instance

Returns the source files audio fourcc compression code.

Audio Levels

A: *<not available>*
D: DWORD __stdcall DTPreview_AudioLevels(void * pPE, long lChannelPair, long * plLeftPeak, long * plLeftRMS, long * plRightPeak, long * plRightRMS);

- pPE – Pointer to preview engine instance
- lChannelPair – which pair of channels to use. 1 = 1/2, 2 =3/4, 3 = 5/6, 4 = 7/8
- lpLeftPeak – peak audio level value for the last video frame 0..65535
- lpLeftRMS – rms audio level value for the last video frame 0..65535
- lpRightPeak – peak audio level value for the last video frame 0..65535
- lpLeftRMS – rms audio level value for the last video frame 0..65535

Returns the peak and RMS levels for the last video frame of audio data.

Free

A: *<not available>*

D: DWORD __stdcall DTPreview_Free(void * pPE, void * pAlloc);

- pPE – Pointer to preview engine instance
- pAlloc – Pointer to memory allocated by DTPreview_GetCurExtendedData or DTPreview_SourceMetaDataSTR

Free memory allocated by DTPreview_GetCurExtendedData or DTPreview_SourceMetaDataSTR, returned via pointer to pointer.

SetMode

A: *<not available>*

D: DWORD __stdcall DTPreview_SetMode(void * pPE, void * pMediaCMD);

- pPE – Pointer to preview engine instance

Send a direct MediaCMD to the preview engine. See the MediaCMD documentation for more information:

http://www.mediacmd.com

Closed Captions and Safe Zone Overlays

Using the SetMode function, closed caption and safe zone overlays can be set on the OpenGL render of the video picture. These overlays will not effect the SDI/HDMI output of a video card.