MOYEA SOFTWARE CO., LTD.

USER'S GUIDE

for Moyea Flash Video MX SDK V2

December 1, 2007
Document version: 2.0

www.moyea.com

TABLE OF CONTENTS

Introduction for Flash Video MX SDK V2

Welcome to Moyea Flash Video MX SDK V2

Moyea Flash Video MX SDK V2 create FLV Flash Video encoded from virtually any video format.

This SDK can be used in server applications to automatically convert uploaded video to FLV, or it can be used to build FLV Flash video conversion into multimedia applications.

Through the built-in Moyea Task Dispatcher, you can easily dispatch the Video to FLV conversion service to the server. More information on Moyea Task Dispatcher, please refer to **User's Guide of Moyea Task Dispatcher**.

The SDK is much more than a simple Encoder - it is a professional tool for the automated creation of Flash streaming video, allowing full control over a wide range of audio and video settings, specified text and image watermark adding，thumbnail picture capturing, and more advanced settings.

You can also convert the FLV files generated from Flash Video MX SDK to SWF files via tools offered by us.

Moyea Flash Video MX SDK V2 provide FLV files ready for the web.

Moyea Flash Video MX SDK V2 helps to realize your ideas on FLV, Flash video by offering the components below:

Moyea Flash Video MX SDK Command Line

You can use Moyea Flash Video MX SDK Command Line on a webservers and run conversion using command line parameters automatically by scheduling or by script calling.

Moyea Flash Video MX SDK COM

Moyea Flash Video MX SDK COM exists in the form of COM component. You can apply this component with any programming language that supports COM component, such as C/C++, VB, Delphi, ASP/ASP.NET and so on.

Features

- **Converts Video and Audio from Virtually Any Format**
  Windows Media (AVI, WMV, ASF, WMA, WAV, using any AVI/WM codecs like DIVX, XDIV, etc.); QuickTime (MOV, QT, DV, AAC, AIF/AIFF, using any QuickTime codecs) and MPEG (MPEG-1, MPEG-2, MPEG-4, MP3).
- **Available for a Wide Range of Programming Languages**
  The SDK is available with two different interfaces: a COM component and a Command line, making it usable from virtually any programming language like VB, VBScript, Delphi, C++, PHP, C#, PERL, etc.
- **Documentation and Examples**
  A number of examples of usage are included in the SDK. Extensive documentation detailing the object methods and properties in PDF format available.
- **Much more than a Simple Encoder**
  The SDK is a very powerful and complete piece of video technology - much more than simple video encoding, Through the cooperation with Moyea Task Dispatcher, The SDK is a complete tool for the automated creation of Flash streaming video, the most flexible and safe way of creating video for today's World Wide Web.
- **Allowed to place a Logo or Image over the Video**
  A solid or transparent image can be placed over the video.
- **Powerful Audio Settings**
  Total replacement of the existing audio track is allowed. Full volume control is also available.
- **Powerful Video Settings**
  Frame rate，bit rate, size, keyframe, brightness, contrast, video fade-in/fade-out and so on.
- **Video/Audio Sequencing**
  Several source video and audio files can be sequenced into a single FLV file. If continue for the SWF file by calling the FLV2SWF tool, then a single SWF file can be arrived at.
- **Convenient screen capturing**
  Allows users to capture frames at random or at a particular time point, or to generate a series of images.
- **Advanced Setting**
  You can set the priority of this process. As to Operation Systems of multiple-processors, the process can be locked to a certain processor.

User's Guide for Moyea Flash Video MX SDK V2 Command Line

Welcome to Moyea Flash Video MX SDK Command Line

Moyea Flash Video MX SDK Command Line offers series of tools to help realize your ideas on FLV(Flash Video).

Via Flash Video MX Command Line tool，you can creates FLV encoded from virtually any video format. You will have Full control over the converted Flash Video: video quality, frame rate, size, duration and other parameters, you can handle parameters using command line.

Via FLV2SWF Command Line tool, you can convert the FLV file generated from Flash Video MX Command Line to SWF file.

Via Picture Grabber Command Line tool, you can capture thumbnail pictures from virtually any video format.

Appendix A introduces some common skills on Command Line.

Flash Video MX Command Line

Function Description

1. Supports video editing, including that of cropping, trimming, merging, deinterlacing, brightness and contrast adjustment, volume control, etc.
2. Could display image watermark and text watermark at any time segment, supporting GIF animations, PNG images with Alpha channel and support to chroma-key the background of the image for further editing. And it supports different stretch mode, transparency, fade-in duration and fade-out duration of the watermark.
3. Rich encoding options, allowing users to specify the dimensions of the output image, stretch mode, output frame rate, audio sample rate, channels, bit rate, fade-in and fade-out, etc.
4. Allows users to capture frames at random or at a particular time point, or to generate a series of images according to a file name template. And users could set the format for the image (JPG/TIFF/GIF/BMP/PNG), cropping, dimension, stretch mode, etc.
5. Supports to generate video only FLV or audio only FLV.
6. Allows users to input instructions through Standard Input to control the encoding process and access the encoding status anytime.
7. Allows users to control the CPU usage of the program during the conversion or encoding.
8. Runs the process on some set of processors.
9. The program provieds more advanced options to control the encoding.

Command Line

flashvideomx [options] outfile[.flv]

outfile

Set the output file name.

If you name the output file as demo or demo.flv, you will get a file named demo.flv.

Option Table

The tables below show all the options that Flash Video MX Command Line supports.

Source options

-sv video_file_name	**-svd**
-sa audio_file_name	**-svc** left:top:right:bottom
-s file_name	**-st** [s|m|e]off_time:duration_time

Video options

-vbr number	**-vfit** time
-vs WxH	**-vfoc** rgb
-vfr number	**-vfot** time
-vgop number	**-vsm** stretch_mode
-vc video_codec	**-vbn** number
-vfic rgb	**-vct** number

Audio options

-asr number	**-afot** time

-abr number	**-ac** audio_channels
-afit time	**-av** number

Logo options

-l logo_file_name	**-lta** number
-ls WxH	**-lfit** time
-la_origin number	**-lfot** time
-la_off x_offset:y_offset	**-lsm** stretch_mode
-ltc rgb	**-lt** [s|m|e]off_time:duration_time
-ltt number	

Text options

-t text_string	**-tfe** number
-ta_origin number	**-tfc** rgb
-ta_off x_offset:y_offset	**-tfa** number
-tff font_name	**-tfit** time
-tf_style number	**-tfot** time
-tf_size number	**-tt** [s|m|e]off_time:duration_time

Grab picture options

-gp file_name	**-gp_size** WxH
-gpw number:file_name_template	**-gp_stretch** stretch_mode
-gpf pic_format	**-gpt** [s|m|e]off_time
-gpc left:top:right:bottom	

Advanced options

-i {v|a}	**-im**
-to time	**-q**
-slp time	**-c** close_stream_type
-p number	**-pa** hex_number
-fi file_name	

Help options

-v	**-h**
-?	

Source options

-sv video_file_name

Add the video specified by the video_file_name.

-sa audio_file_name

Add the audio specified by the audio_file_name.

-s file_name

Add the video/audio specified by the file_name.

-svd

Set modified the deinterlace video or not.

if the input file is the interlaced video, it will be modified, if not, the option will be canceled.

-svc left:top:right:bottom

Set the area of the input video to be cropped.

The pixels of each lateral to be cropped away. This combination of pixels will form a selected content. For instance, 0:20:0:20 indicates that you are to crop away 20 pixels from both the top and the bottom.

-st [s|m|e]off_time:duration_time

Set the time segment of the video/audio file that is to be trimmed (millisecond).

duration_time indicates the time that lasts (the value is positive).

off_time inidcates offset value of the the time. It may have prefixes as s, m, e.

When the off_time has no prefixes, the default offset value is set as the start time of the video.

The prefix "s" indicates the start place of the video and the offset value is positive, e.g. -st s1000:60000, -st 0:60000;

the prefix "m" indicates the middle place of the video. The middle and the rightward offset value is positive, and the middle and leftward offset value is negative, e.g. st m1000:60000, -st m-1000:60000;

the prefix "e" indicates the finish place of the video, and the offset value is negative, e.g. -st e-1000:60000.

this option may appear repeatedly.

When this option is canceled, all items will be added by default.

Video options

-vbr number

Set the video bit rate (bps).

“number” is the value of the bit rate in bps. If you want to set 400kbps as the bit rate, it should be like this: -vbr 400000.

The default is 400000.

-vs WxH

Set the size of the Video (pixel). e.g. -vs 320x240.

W and H must be even numbers.

When this option is not specified, the width and height of the output FLV file are decided by that of the first video source that is added to the Encoder object.

-vfr number

Set the frame rate of the video. For example, if you want to specify the video frame rate as 29.97, you can specify: -vfr 29.97.

When this option is not specified, the frame rate of the output FLV file is decided by that of the first video source that is added to the Encoder object.

-vgop number

Set the space between the keyframes. e.g.: -vgop 12.

The default value is 12.

-vc video_codec

Set the video encoder; and you have the following options:

H263 H263 encoding

VP6 VP6 encoding

The default is *H263*.

-vfic rgb

Set the color value of the video fade-in.

"rgb" is an 6-digit hexadecimal integer with the form of rrggbb; and each component occupies two hexadecimal digits. For example, -vfic FF0000.

it's recommended that it should be used together with **-vfit** time, like -vfic FF0000 -vfit 5000.

-vfit time

Set the fade-out time of the video (millisecond).

it's recommended that it should be used together with **-vfic** rgb, like -vfic FF0000 -vfit 5000.

-vfoc rgb

Set the color value of the fade-out.

"rgb" is an 6-digit hexadecimal integer, like the form of rrggbb; and each component occupies two hexadecimal digits. For example, -vfoc FF0000.

It's recommended that it should be used together with**-vfot** time, like-vfoc FF0000 -vfot 5000

-vfot time

Set the video fade-out time (millisecond).

It's recommended that it should be used together with**-vfoc** rgb, like -vfoc FF0000 -vfot 5000

-vsm stretch_mode

Set the stretch mode of the video, possible value (not case sensitive):

stretch	The image is stretched to full the screen, without preserving the aspect ratio.
letterbox	The image is resized to fit the screen in one dimension, width or height, while preserving the aspect ratio. If the ratio of width to height in the image does not match the ratio in the target frame, it creates a letterbox.

The default is stretch.

-vbn number

Set the brightness of the video, -127~127。

-127	the less bright
0	do not alter the original brightness
127	the most bright

The default is 0.

-vct number

Set the contrast of the video, -127~127

-127	the weakest contrast
0	do not alter the original contrast
127	the strongest contrast

The default is 0.

Audio options

-asr number

Set the audio sample rate, possible value:

11025	11025Hz
22050	22050Hz
44100	44100Hz

The default value is 44100.

-abr number

Set the audio bit rate.

"number" is the value of the audio bit rate in bps. If you want to set 96kbps as the audio bit rate, it

should be like this: -abr 96000.

The possible values for Sample Rate and Audio Bit Rate are listed as follows:

Sample Rate(Hz)	Audio Bit Rate(bps)
11025	16000,32000,40000,48000,56000,64000,80000,96000,112000, 128000,160000
22050	16000,32000,40000,48000,56000,64000,80000,96000,112000, 128000,160000
44100	32000,40000,48000,56000,64000,80000,96000,112000,128000, 160000,192000,224000,256000,320000

The default value is 96000.

-afit time

Set the audio fadein.

"time" is the time for audio to fade in; it measures in milliseconds. If you want to set 1000 milliseconds as the time to fade in, it should be like this: -afit 1000.

-afot time

Set the audio fadeout.

"time" is the time for audio to fade out; it measures in milliseconds. If you want to set 1000 milliseconds as the time to fade out, it should be like this: -afot 1000

-ac audio_channels

Set the audio channel; possible value (not case sensitive):

mono monophonic

stereo standard stereo

The default is stereo.

-av number

Adjust the volume. 0~4.

If you want to turn up the volume to two times of the original, you can set : -av 2

If you want to turn down the volume to half of the original, you can set: -av 0.5.

The default value is 1.

Logo options

-l logo_file_name

Set the file name of the Logo image.

-ls WxH

Set the size of the Logo (pixel), e.g. -ls 32x24.

The default size is 32x24.

-la_origin number

Set the relative original point of the Logo.

0 Top-Left

1 Top-Center

2 Top-Right

3 Center

4 Bottom-Left

5 Bottom-Center

6 Bottom-Right

7 Left-Center

8 Right-Center

The default value is 0.

-la_off x_offset:y_offset

Set the x,y offset values of the Logo.

x_offset, y_offset indicate the horizontal offset and the vertical offset respectively.

Horizontally right and vertically down are positive values, and horizontal left and vertically up are negative value. e.g. -la_off 30:-40.

The default value is 0.

-ltc rgb

Set the transparent color of the Logo.

"rgb" is an 6-digit hexadecimal integer with the form of rrggbb; and each component occupies two hexadecimal digits. For example, -ltc FF0000.

The default is non-transparency.

-ltt number

Set the tolerance value for chroma keying on the Logo,

"number" is the tolerance value,any pixels on the Logo that have a similarity degree with the transparent color between 0 and the tolerance value will be transparent.

The default is 0.

-lta number

Set the transparency of the Logo. 0~255.

0 totally transparent

255 totally no transparency

The default is 200

-lfit time

Set the fade-in time of the Logo (millisecond).

-lfot time

Set the fade-out time of the Logo (millisecond).

-lsm stretch_mode

Set the stretch mode of the Logo, possible value (not case sensitive):

none do not stretch

center do not stretch and set the Logo at the center

stretch stretch the Logo

tile set the Logo as tiled

The default is stretch.

-lt [s|m|e]off_time:duration_time

Add a time segment for the Logo to show(millisecond).

duration_time indicates the time that lasts, the value is positive.

off_time indicates offset value of the time. It may have such prefixes as s, m, e.

when the off_time has no prefixes, the offset value is related to the start of the video.

The prefix "s" indicates the origin is the start of the video and the offset value is non-negative, e.g. -st s1000:60000, -st 0:60000

the prefix "m" indicates the origin is the middle of the video. The middle and the rightward offset value is positive, and the middle and leftward offset value is negative, e.g. st m1000:60000, -st m-1000:60000

the prefix "e" indicates the origin is the end of the video, and the offset value is negative, e.g. -st e-1000:60000.

This option may appear repeatedly.

When this option is not specified, the Logo will be displayed from the beginning to the end.

Text options

-t text_string

Add a text watermark.

text_string is the text string to be shown.

-ta_origin number

Set the relative original point of the Text.

0 Top-Left
1 Top-Center
2 Top-Right
3 Center
4 Bottom-Left
5 Bottom-Center
6 Bottom-Right
7 Left-Center
8 Right-Center

The default is 0

-ta_off x_offset:y_offset

Set the x,y offset values of the Text.

x_offset, y_offset indicate the offset value on the horizontal direction and vertical direction.

it is preset that horizontally right and vertically down is positive; and horizontally left and vertically up is negative.

The default is 0.

-tff font_name

Set the font name, e.g. "MS Sans Serif".

The default is "Times New Roman".

-tf_style number

Set the font style, possible values:

0 Normal
1 Italic
2 Bold
3 Bold Italic

The default is 0.

-tf_size number

Set the font size

The default is 20.

-tfe number

Set the font effect, possible values:

0 None
1 Strike out
2 Under line
3 Strike out，Under line

The default is 0.

-tfc rgb

Set the font color.

"rgb" is an 6-digit hexadecimal integer with the form of rrggbb; and each component occupies two hexadecimal digits. For example, -ltc FF0000.

the default value is D7BC02.

-tfa number

Set the font transparency. 0~255.

0 totally transparent

255 no transparency

the default is 255.

-tfit time

Set the fade-in time of the text watermark (millisecond).

-tfot time

Set the fade-out time of the text watermark(millisecond).

-tt [s|m|e]off_time:duration_time

Add a time segment for the Text (millisecond).

duration_time indicates the time that lasts (the value is positive).

off_time indicates offset value of the time. It may have prefixes as s, m, e.

When the off_time has no prefixes, offset value is related to the start of the video.

The prefix "s" indicates the origin is the start of the video and the offset value is non-negative, e.g. -st s1000:60000, -st 0:60000;

the prefix "m" indicates the origin is the middle of the video. The middle and the rightward offset value is positive, and the middle and leftward offset value is negative, e.g. -st m1000:60000, -st m-1000:60000;

the prefix "e" indicates the origin is the end of the video, and the offset value is negative, e.g. -st e-1000:60000.

This option may appear repeatedly.

When this option is not specified, the Text will be displayed from the beginning to the end.

Grab picture options

-gp file_name

Set the file name for the captured image.

-gpw number:file_name_template

Grab number images and store them with the name as specifed by file_name_ template. The file_name_template string must contain “?”, and this “?” will be replaced by the number when the files are generated.

for example, with -gpw 5:demo??.bmp, the image series will be demo01.bmp, demo02.bmp,…, demo05.bmp.

-gpf pic_format

Set the output grabbed image format, jpg/tiff/gif/bmp/png.

-gpc left:top:right:bottom

Set the area to crop.

The pixels of each lateral to be cropped away. This combination of pixels will form a selected content, e.g. 0:20:0:20 indicates that 20 pixels will be cropped away from both the top and the bottom.

When not specified, no cropping.

-gp_size WxH

Set the size of the grabbed output (pixle). Like -gp_size 320x240.

When not specified, the size of the grabbed will be the width and height of the output FLV file.

-gp_stretch stretch_mode

Set the stretch mode of the grabbed image, possible value (not case sensitive):

stretch The image is stretched to full the screen, without preserving the aspect ratio.

letterbox The image is resized to fit the screen in one dimension, width or height, while preserving the aspect ratio. If the ratio of width to height in the image does not match the ratio in the target frame, it creates a letterbox.

The default is stretch.

-gpt [s|m|e]off_time

Set the time for grabbing image (millisecond).

off_time indicates the time offset value, and it may have these prefixes: s, m, e.

When the off_time has no prefixes, the default offset value is set as the start time of the video.

The prefix "s" indicates the origin is the start of the video and the offset value is non-negative, e.g. -gpt s1000, -gpt 2000;

the prefix "m" indicates the origin is the middle of the video. The middle and the rightward offset value is positive, and the middle and leftward offset value is negative, e.g.-gpt m1000, -gpt m-1000;

the prefix "e" indicates the origin is the end of the video, and the offset value is negative, e.g.-gpt e-1000.

When this option is not specified, you are set to grab image at random.

Advanced options

-i {v|a}

When the option is “v”, the flv file will only contain video data.

When the option is “a”, the filv file will only contain audio data.

-to time

Set the time to deadlock after some seconds. The deadlock time means that if there is no conversion for video data or audio data in the number of time, the deadlock status is enabled. If you want to set 60 seconds as the deadlock time, it should be like this: -to 60.

The default deadlock time is 60s.

-slp time

Set the time for sleep after encoding each frame, in milliseconds. It is used to prevent the program from increasing CPU usage during the process of encoding. By default, the value is 0, which indicates not to sleep, so that the encoding will be finished as soon as possible. Like -slp 50.

The default is no sleep.

-p number

Set the priority of this process.

0 normal

1 below normal

2 low

-fi file_name

Show the video and/or audio information of the file specified by file_name, then exit.

-im

Use interaction mode.

In this mode, you can issue the following control instructions:

gdt	get the total time of the encoding
gct	get the current progress of the encoding
gvbr	get the video bit rate
gabr	get the audio bit rate
run	go on to encode
pause	suspend the encoding
stop	stop the encoding
terminate	forcefully end the encoding and exit the program
isfinish	see if the encoding is finished or not. If the encoding is finished, display "yes"; if not, display "no"

When it's in interaction mode, all the control instructions above are read from Standard Input, which is keyboard by default, and all the results are write to Standard Output, which is screen by default. However, you can redirect them to files or pipes.

More information, please refer to **About interaction mode**.

-q

Use silent mode, not to display the progress information.

-c close_stream_type

Close the specified output stream.

stdout

close standard output stream.

stderr

close standard error stream.

alloutput

close standard output stream and standard error stream.

notclose

do not close the output stream.

By default, standard output stream and standard error stream are printed to the screen.

-pa hex_number

Runs the process on some set of processors.

Binary bit 0 stands for processor 0, while bit 31 stands for processor 31.

-pa 0x5, 0x5 appears as 101 under binary, and the bit 0 and bit 2 are 1, others are 0, showing that the current process can run on CPU 0 and CPU 2, but can not run on CPU 1 and CPU 3 or above to CPU 31.

Help options

-v

Show the version info.

-?

Show help info.

-h

Show help info.

Exit Code

- 1	Unhandled exception
0	The program has successfully completed the relevant operation and returns back
1	Parameter error
2	Has not added any file
3	Failure to open the FLV video encoder
4	Failure to open the FLV audio encoder
5	The file is running in other program
6	Error to write in the file
7	Memory allotment failure
8	Failure to inject FLV meta information
9	Failure to get the standard output handle
10	Deadlock when adding source
11	Has not specified the file name when getting the video/audio information
12	Deadlock when encoding video/audio
13	User issues the terminate instruction in the interactive mode to end the encoding
14	Memory mapping file error
15	Failure to create encoder
16	Failure to create file mapping object
17	Moyea Task Executer memory mapping file version does not match
18	Moyea Task Executer issues stop instruction to exit the program
19	Moyea Video2FLV.dll issues terminate instruction to exit the program
20	Moyea Video2FLV.dll fails to create encoder
21	Unmatched Sample Rate and Audio Bit Rate
22	Failed to set the running processor set of the process

Samples

1. Convert a video file to FLV with original dimension and frame rate:
   flashvideomx -s myvideo.avi myflash
   This command is to convert myvideo.avi to myflash.flv.

2. Convert a video file to FLV with settings about output dimension, video bit rate, frame rate, audio sample rate, audio bit rate and audio channel:
   flashvideomx -s myvideo.avi myflash -vs 320x240 -vbr 400000 -abr 64000 -asr 22050 -ac mono -vfr 25
   This command is to convert myvideo.avi to myflash.flv with 320x240 as dimension, 400kbps as video bit rate, 64kbps as audio bit rate, 22050Hz as audio sample rate, monophonic channel and 25fps as frame rate.

3. Add a LOGO image to video:
   flashvideomx -s myvideo.avi myflash -l "mylogo.jpg" -ls 100x100 -la_origin 0 -la_off 10:10 -lt 0:10000 -lt e-10000:10000
   This command is to stretch mylogo.jpg to 100x100 in dimension and show the image when the video starts to play and last 10000 milliseconds, and show the image again when there are still 10000 milliseconds to the end of the video, and last 10000 milliseconds with the coordinate (10,10).

   flashvideomx -s myvideo.avi myflash -l "mylogo.jpg" -ls 100x100 -la_origin 0 -la_off 10:10
   This command is to stretch mylogo.jpg to 100x100 in dimension and show the image from the beginning to the end of the video with the coordinate (10,10).

4. Add characters to video:
   flashvideomx -s myvideo.avi myflash -t "Text to show on video" -ta_origin 0 -ta_off 0:0 -lt 0:10000 -lt e-10000:10000
   It is to show “Text to show on video” in the top left corner and show the characters when the video starts to play and last 10000 milliseconds, and show the characters again when there are still 10000 milliseconds to the end of the video, and last 10000 milliseconds.

   flashvideomx -s myvideo.avi myflash -t "Text to show on video" -ta_origin 0 -ta_off 0:0
   -tff "Times New Roman" -tf_style 0 -tf_size 24 -tfe 0 -tfc FF0000 -tfa 80
   It is to show the semi-transparent characters “Text to show on video” in the top left corner with “Times New Roman” as font, “24” as size, “FF0000” as color (red).

5. Connect several videos into one file:
   flashvideomx -s headervideo.avi -s myvideo.avi -s tailervideo.avi myflash
   It is to connect headervideo.avi, myvideo.avi, and tailervideo.avi together and convert them to one file: myflash.flv

6. Replace the audio data in the video file:
   flashvideomx -sv myvideo.avi -sa myaudio.mp3 myflash
   It is to replace the audio data in myvideo.avi with myaudio.mp3, and then convert to myflash.flv.

7. No audio:
   flashvideomx -s myvideo.avi myflash -i v

This command is to generate an FLV file without audio data.

8. No Video:
   flashvideomx -s myvideo.avi myflash -i a
   This command is to generate an FLV file without video data.

9. Get thumbnail picture:
   flashvideomx -s myvideo.avi myflash -gp mypic.jpg -gp_size 120x120
   This command is to convert myvideo.avi to myflash.flv, and generate a random mypic.jpg in size 120x120 as the thumbnail picture.

   flashvideomx -s myvideo.avi myflash -gp mypic.jpg -gp_size 120x120 -gpt 60000
   This command is to convert myvideo.avi to myflash.flv, and generate pictures in mypic.jpg with the size of 120x120 in dimension in the given 60000 milliseconds.

   flashvideomx -s myvideo.avi myflash -gpw 10:mypic??.jpg -gp_size 120x120
   This command is to convert myvideo.avi to myflash.flv, and capture 10 pictures in 120x120 dimension as: mypic01.jpg, mypic02.jpg, ..., mypic10.jpg.

10. Convert the video to FLV with time trim:
    flashvideomx -s myvideo.avi -st 0:300000 myflash
    This command is to convert myvideo.avi with 300000 milliseconds from the starting point of the video into myflash.flv.

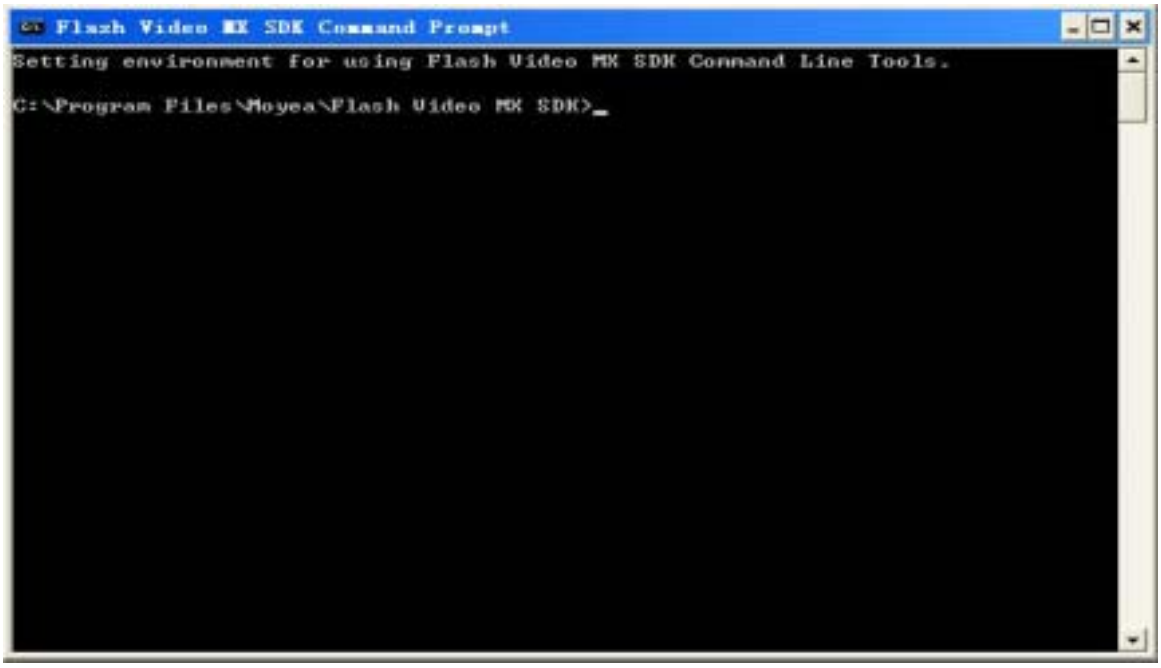
    flashvideomx -s myvideo.avi -st 0:300000 -st e-300000: 300000 myflash
    This command is to convert the two video parts in myvideo.avi to myflash.flv. One is the part from the start point to the 300000ms point. The other part is the one from the point 300000ms before the end point to the end point.

About interaction mode

Through the interaction mode, you can look up the current encoding status of Flash Video MX and control its encoding process.

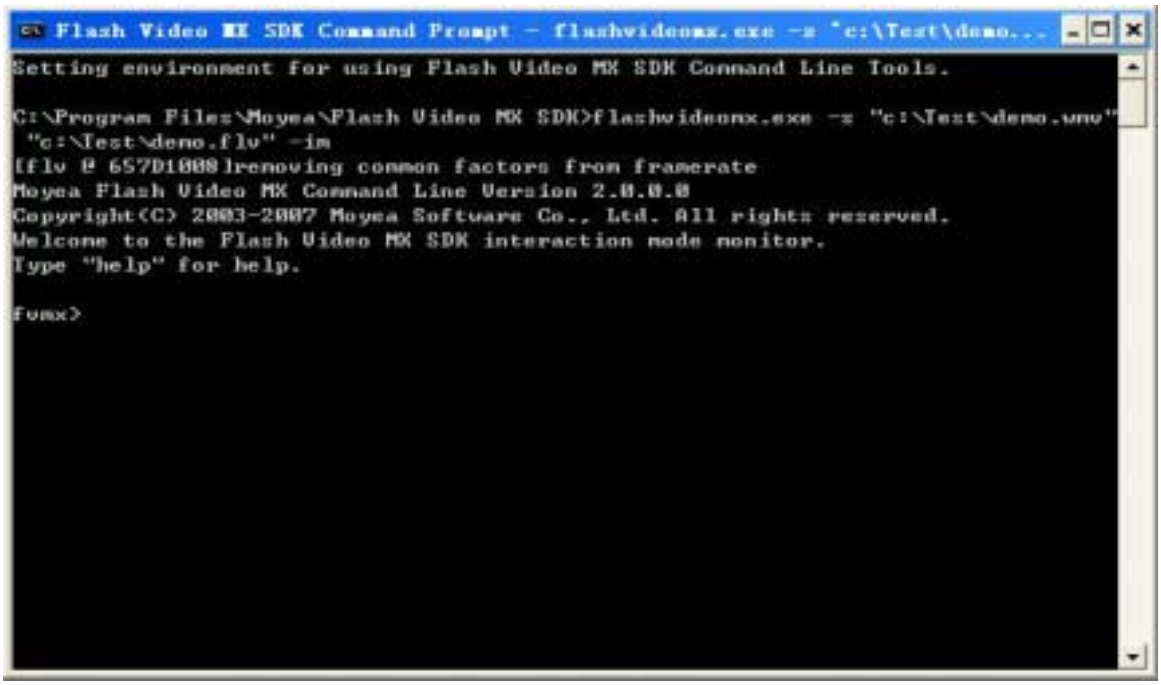
If worked under interaction mode, you can follow the steps shown below.

1. Open the Command prompt of Flash Video MX SDK, as shown below:

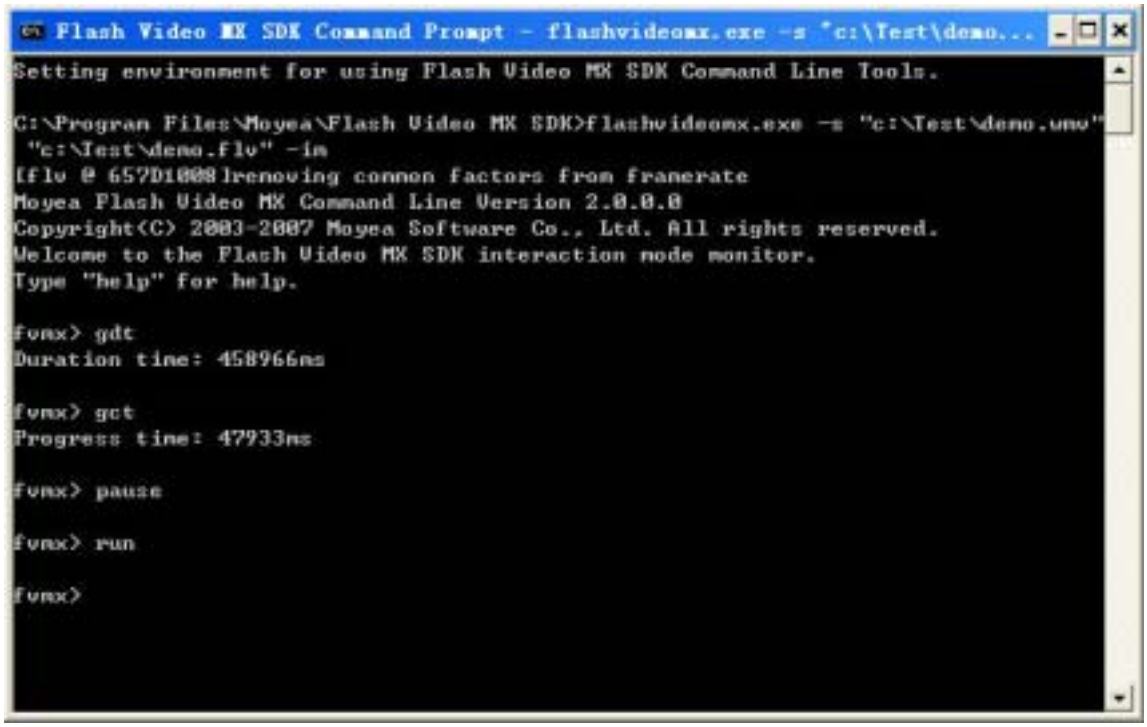


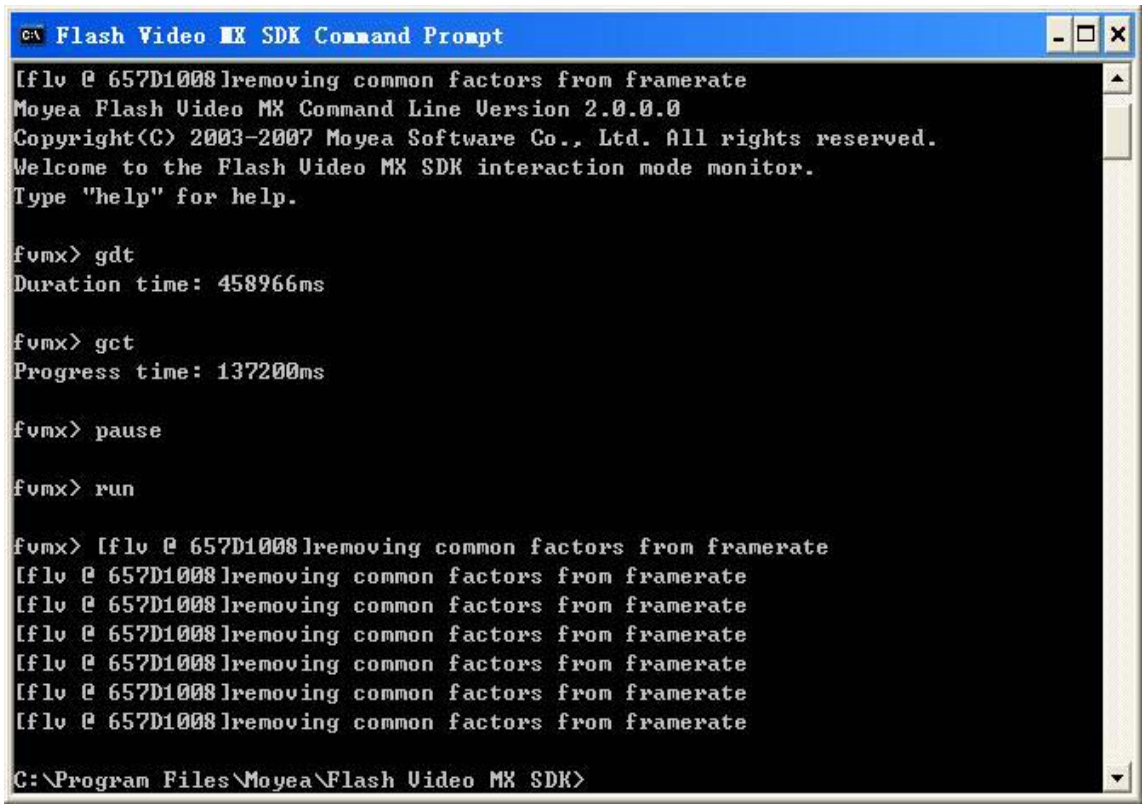
2. Execute a command with -im, and the application will start to encode under interaction mode.

For example: flashvideomx.exe -s "c:\Test\demo.wmv" "c:\Test\demo.flv" -im



3. Execute the commands like gdt, gct, gvbr, etc. When the encoding finished, it returns to the Flash Video MX SDK Command prompt, as shown below:





FLV2SWF Command Line

Function Description

Convert the FLV file generated by Flash Video MX SDK to SWF file.

Command Line

FLV2SWF flv_file swf_file [options ...]

flv_file

input path of FLV.

swf_file

output path of SWF.

Option Table

The table below shows all the options that FLV2SWF Command Line supports.

-ver:x	**-bt:**x
-bgc:x	**-br:**x
-stop_at_start	**-bb:**x
-stop_at_end	**-vw:**x
-rewind	**-vh:**x
-nosound	**-h|-?**
-bl:x	

Options are:

-ver:x

set the file version of the output SWF file, x can be 7 or 8.

-bgc:x

set the background color in $rrggbb format, 'none' for transparent color.

e.g. -bgc:$FF0000, indicates red background;

-bgc:none, indicates transparent background.

-stop_at_start

place a stop action in the first frame.

-stop_at_end

place a stop action in the last frame.

-rewind

place a gotoAndPlay(1) action in the last frame.

-nosound

drop sound stream.

-bl:x

set the width of the left border in pixel.

-bt:x

set the height of the top border in pixel.

-br:x

set the width of the right border in pixel.

-bb:x

set the height of the bottom in pixel.

-vw:x

set the width of the video in pixel.

-vh:x

set the height of the video in pixel.

-h|**-?**

show the help info.

Exit Code

0 is success; other values are failure.

Picture Grabber Command Line

Function Description

Generate thumbnails from the source video file.

Command Line

PictureGrabber [options ...]

Option Table

The table below shows all the options that Picutre Grabber Command Line supports.

-i source	**-gpw** number:file_name_template ...
-gpf pic_format	**-gp** file_name [**-gpt** [s|m|e]off_time] ...
-gpc left:top:right:bottom	**-fi** source
-gp_size WxH	**-v**
-gp_stretch stretch_mode	**-h**|**-?**

Options are:

-i source

Set the source file to generate thumbnails.

-gpf pic_format

Set the file format for the output images. jpg, tiff, gif, bmp or png.

-gpc left:top:right:bottom

Set the region to be left.

The pixels of each lateral are to be cropped away. This combination of pixels will form a desired content, e.g. 10:20:30:40 indicates that 20 pixels will be cropped away from the left lateral, 20 pixels from the top lateral, 30 pixels from the right lateral and 40 pixels from the bottom lateral.

When not specified, no crop.

-gp_size WxH

Set the size of the image (in pixel). -gp_size 320x240 indicates the output image width is 320 pixels, and height is 240 pixels.

-gp_stretch stretch_mode

Set the stretch mode of the image with possible value (not case sensitive):

stretch The image is stretched to fit the output frame size without preserving aspect ratio. (Default)

letterbox The original aspect ratio is preserved, with a black letterbox along the shorter dimension.

-gpw number:file_name_template ...

Grab images in number and save with the name of file_name_template.

The file_name_template string must contain “?”, which will be replaced by the number when the images are generated.

For example, with -gpw 5:"C:\demo??.bmp", the image series will be C:\demo01.bmp, C:\demo02.bmp,…, C:\demo05.bmp.

-gp file_name [**-gpt** [s|m|e]off_time] ...

Generate a thumbnail image at a given time.

-gp file_name

Name the captured image.

-gpt [s|m|e]off_time

Set the time for grabbing images (millisecond).

off_time indicates the offset value of the time, which may contain these prefixes: s, m, e.

When the off_time contains no prefixes, the default offset value is the video start time.

The prefix "s" indicates the origin of the image grabbing time is the video start and the offset value is non-negative, e.g. -gpt s1000.

The prefix "m" indicates the origin of the image grabbing time is the video middle. The middle and the rightward offset value are positive, e.g. -gpt m1000; while the middle and leftward offset value are negative, e.g. -gpt m-1000.

The prefix "e" indicates the origin of the image grabbing time is the video end, and the offset value is negative, e.g. -gpt e-1000.

If this option is not specified, images will be grabbed at random.

-fi source

Show the video and/or audio information of the source, then exit.

-v

Show the version info.

-h|**-?**

Show the help info.

Samples

PictureGrabber -i demo.avi
-gpc 12:20 -gp_size 1024x768 -gp_stretch letterbox
-gpw 5:"C:\test1??.jpg"
-gpw 5:"C:\test2??.tiff"
-gp "C:\a.gif" -gpt s3000
-gp "C:\b.bmp" -gpt m-4000
-gp "C:\c.png" -gpt e-5000
-gp "C:\d.jpg"

Generate thumbnails from demo.avi, and crop away 12 pixels from the left lateral, 20 pixels from the top lateral. The generated image size is 1024x768 pixels, and the stretch mode is letterbox. Generate thumbnails, C:\test101.bmp, C:\test102.bmp, … C:\test105.bmp and C:\test201.bmp, C:\test202.bmp, …, C:\test205.bmp at random. Generate the thumbnail C:\a.gif at the 3000th ms, the thumbnail C:\b.bmp at the time point 4000ms before the video middle, the thumbnail C:\c.png at the time point 5000ms before the video end, and the thumbnail C:\d.jpg at random.

Exit Code

0	success
1	error on parameters
2	fail to get the info from the source file
3	fail to select source

Appendix A General tips on Command Line

1. Open the Command Prompt
a. Begin with **start**/**Run**, type in **cmd**, and click **OK** or press **Enter**
b. Begin with **start**/ **Programs**/**Accessories**/**Command prompt**
c. Press **Win**+**R**, and the **Run** window will pop up, type in **cmd**, click **OK** or press **Enter**

2. Change drives, e.g. from the current drive to drive D, type in **d:**

3. Display the a list of files and sub-directories under the current directory, type in **dir**

4. Change the current directory, type in **cd path**; return to the parent directory, type in **cd ..**
; back to the root directory, type in **cd **

5. Open the directory through the **Explorer**, type in **start path**. Specifically open the current directory, type in **start .**

6. Get the previous commands, press ↑ and ↓ to choose

7. File and directory name automatically filled up. Type in the initial one or several letters and press **Tab**

8. Get the exit code from the command line and confirm whether the process is successful, type in **echo %errorlevel%**

9. Displays a search path for executable files, type in **path**
Sets a search path for executable files, type in **path [[drive:]path[;...][;%path%]**
e.g. Type in **path c:\test;%path%**, and c:\test will be added to the current search path

10. Start another command prompt, keeping with the current settings, type in **start cmd**

11. Clear the screen，type in **cls**

User's Guide for Moyea Flash Video MX SDK V2 COM

Welcome to Moyea Flash Video MX SDK COM

Flash Video MX SDK COM exists in the form of COM component. You can apply this component with any programming language that supports COM component, such as C/C++, VB, Delphi, ASP/ASP.NET and so on.

Flash Video MX SDK COM helps to realize your ideas on FLV(Flash Video) via the cooperation of the different objects offered below.

Encoder Object

Via this Object， you can create FLV files from from virtually any video format. You are provided with the full control over the generated Flash Video: quality, frame rate, size, and duration, etc. You can handle the parameters via the properties or methods of the object.

EncodingTask Object

If you intend to execute the code in the task mode, then you can look up the information of the encoding status via EncodingTask Object, like: video bite rate, current encoding time, encoding result, etc.

VideoInfo Object

Via this Object, you can obtain the information of the video and/or audio.

License Object

When you deploy your program, you need to use License Object to register Flash Video MX SDK first.

FLV2SWF Object

This Object is to convert the FLV file generated by Flash Video MX SDK to SWF file.

PictureGrabber Object

Via this Object, you can capture screens of the source files.

Introduction

Flash Video MX COM Edition lets you create video to FLV application very easily in the WEB environment.
Flash Video MX COM Edition exists in the form of COM component. You can apply this component with any programming language that supports COM component, such as C/C++, VB, Delphi, ASP/ASP.NET and so on.

Encoding Track

Flash Video MX COM Edition introduces an “encoding track” concept, which means the video or the audio queues, waiting for encoding. There are two encoding tracks, one is for video, and the other is for audio. You can call AddAVSource to add a segment of video to “video encoding track” and a segment of audio to “audio encoding track”, or you call AddVideoSource or AddAudioSource to add a segment of video or a segment of audio separately. When encoding, all the segments in one “encoding track” will be joined together.
If the duration of “video encoding track” and “audio encoding track” are not the same, the shorter one will be padded with black frames or silent sound. With this “encoding track” concept, developers can cut several videos and join them into a new one, or replace the original sound of the video with another sound.

Two Styles of Encoding

Flash Video MX COM Edition introduces two styles of encoding, one is **Encode**, and the other is **EncodeAsTask**. With **Encode**, you can launch encoding immediately after you have specified the encoding parameters, however, the caller have to wait for the encoding process, although the **Encode** does not block the execution of the caller’s thread. However, with **EncodeAsTask**, the encoding is submitted as a task and a task id is returned to the caller, the task will be executed with **Moyea Task Dispatcher** service process, so the caller’s thread does not need to wait for the encoding process. **EncodeAsTask** is designed for the web application, such as ASP/ASP.NET, which always has a time limit for scripts execution. The **EncodeAsTask** returns as soon as possible, so the page execution will not be blocked.
Note: If you want to call **EncodeAsTask** method, you have to start Moyea Task Dispatcher service first.

Encoding Under Another Account

Flash Video MX COM Edition v1.5.0.0 introduces a mechanism to let you launch the encoding process under another account. This only affects the encoding process launch by **Encode**. The encoding process launched by **EncodeAsTask** always runs under the system account. In IIS, the web page is executed under IWAM_XXX (IIS4, IIS5), or Network Service (IIS6), however, the account does not have enough permissions to do encoding and it causes encoding failure. So when you are calling **Encode** in IIS environment (ASP/ASP.NET/ISAPI/CGI), **you need to specify an account in Administrators group to do the encoding** in this case.

Data Type Using

Time

The time here we used is an integer, in millisecond, which is 1/1000 of one second. We use time to specify a start point and duration to cut a segment of video or audio.

Coordinates

The coordinates here we used is an integer, in pixel. The origin is on top-left corner, the positive directions are the right and the bottom, while the negative directions are the left and the top.

Color

The color here we used is an integer in hexadecimal. It's RGB based with rrggbb or bbggrr format.
In C/C++, Javascript, C#, Java, it's 0xrrggbb or 0xbbggrr.
In VB/VBScript, it's &Hrrggbb or &Hbbggrr.

Alpha

Here alpha is the alpha component of a pixel. It describes the pixel's transparency, which ranges from 0 to 255, where 0 is invisible and 255 is opaque.

BitRate

The bitrate here we used is an integer in bps (bits per second). It describes the output video or audio data rate. With the same encoder and the same image, the higher the bitrate is, the better the quality of the image will be. In some case, we use kbps as unit, and 1 kbps = 1000 bps.

What's New

v1.0.0.4

1. Added AddVideoSource and AddAudioSource methods to Encoder object for free video & audio arrangement on target encoding video & audio tracks.
2. Fixed the bug of AddAVSource method, which ignored the pathAudio parameter.

v1.5.0.0

1. Fixed memory leaking in Flash Video MX Server service and COM object.
2. Fixed optional parameters in AddImageWaterMark and AddTextWaterMark, which you may not miss these optional parameters before.
3. Updated Video Drawer component, and now image watermark with alpha channel is supported. And introduced an advanced color chroma keying algorithm to chroma key the background of the watermark.
4. Added UserName and Password properties to Encoder object and VideoInfo object. You can run encoding process under another account.

v2.0.0.0

1. Added AddVideoSourceWithCrop function to Encoder object for adding video source with cropping.
2. Added SetWaterMarkFadeInOut function to Encoder object for setting watermark fading duration.
3. Added TaskCompleteCallBack property to Encoder object to replace EncodingOKUrl and EncodingFailedUrl.
4. Added ThumbnailFormat to Encoder object for setting thumbnail image format obviously.
5. Added Brightness & Contrast to Encoder object for adjusting the video brightness and contrast.
6. Added Volume property to Encoder object to raise or lower the sound in target video.
7. Added ThumbnailStretchMode to Encoder object for setting stretching method used for resizing the thumbnail image.
8. Added EncodingResult to EncodingTask object for getting task encoding result.
9. Added RemoveTask to EncodingTask object for removing task queued.
10. Added DataRate to VideoInfo object for getting average data rate of the video.

v2.0.0.0 made some changes that do not compatible with the old version:

1. The EncodingOKUrl property and EncodingFailedUrl property are obsolete, use the TaskCompleteCallBack instead.
2. EncodingResult returned is different from the old version.
3. EncodeAsTask does not submit the task to Flash Video MX Server service, it, in fact, submits the task to Moyea Task Dispatcher service. For more information, please read the **Flash Video MX SDK Architecture** Document.

Getting Started

1. Create fmx.Encoder object

```
DIM Encoder
SET Encoder = Server.CreateObject("fmx.encoder")
```

2. Set encoding parameters

You can select the source file, set the output settings and the parameters of the output file like the following examples:

```
'set source files
Encoder.AddAVSource("c:\myvideo.avi", "c:\myvideo.avi")
'set output file
Encoder.Output = "c:\myvideo"
'set output parameters
Encoder.Width = 400;
Encoder.Height = 300;
```

3. Execute encoding immediately

```
'Start encoding immediately
Encoder.Encode
'Wait until the encoding completes
Encoder.WaitForEncodingComplete(-1)
```

4. Execute encoding as a task

You can also save the encoding task to the queue and the task monitor will pop the task and execute encoding asynchronously.

```
Encoder.EncodeAsTask
```

Note: If you want to call this method, you have to start Moyea Task Dispatcher service first.

5. Inquire encoding state

1) When you are encoding with "Encode", you should modify the code in step 3 like the following to inquire the encoding state:

```
'Start encoding immediately
Encoder.Encode
'Loop to get encoding statistics
While Encoder.IsEncoding
   'Get statistics
   Encoder.totalTime 'Get duration in ms
```

```
    Encoder.currentTime ‘Get current time in ms
    Encoder.realVideoBtr ‘Get real video bitrate
    Encoder.realAudioBtr ‘Get real audio bitrate
    ‘Wait a timeslice instead of infinite waiting
    Encoder.WaitForEncodingComplete(200)
  Wend
```

2) If you are encoding with EncodeAsTask, you should use EncodingTask object to inquire the task.

```
DIM Id
Id = Encoder.EncodeAsTask
‘Create EncodingTask object
DIM task
SET task = Server.CreateObject(‘fmx.encodingtask’)
task.TaskID = Id
‘Get task state, 0 for waiting, 1 for converting, 2 for finished, -1 for failed.
Response.Write(task.State)
‘Get current time stamp of converted frames in milliseconds
Response.Write(task.currentTime)
‘Get output file duration in milliseconds
Response.Write(task.totalTime)
Response.Write(task.realVideoBtr)
Response.Write(task.realAudioBtr)
```

6. Use callback URL

In the properties of Encoder object, TaskCompleteCallBack is the URL that will be accessed when encoding task is completed, failure or success. Before TaskCompleteCallBack is accessed, the task is still in the task queue. However, once the encoder finished encoding and has accessed the callback URL, the task will be deleted.

7. Get Video Information

Using VideoInfo object, you can get information, like video size, frame rate, duration, audio sample rate, channels.

```
DIM Info
SET Info = Server.CreateObject(“fmx.VideoInfo”)
If (Info.LoadVideo(“xxx.avi”)) then
    Info.Width
    Info.Height
    Info.FrameRate
    Info.SampleRate
    Info.Duration
    Info.Seekable
    Info.AudioChanels
    Info.NeedDeinterlace
End If
```

Samples

1. fmx.Encode/fmx.EncodeAsTask testing script

This is an HTA program used locally. Using Jscript, it can convert local video files to FLV files and show the process of conversion. Please find "encoder_test" folder in the "Samples" folder under the installation folder for Flash Video MX SDK. Open the folder and then double-click "Encode.hta"/"EncodeAsTask.hta" file to run it. You can directly open the "Encode.hta"/"EncodeAsTask.hta" file with a text editor to view the source code for the program.

2. C++ Demo

COM_CUI

COM_CUI is a Visual C++ 6.0 project for a console application to encode a source video into FLV.

COM_GUI

COM_GUI is a Visual C++ 6.0 project for a desktop application to encode a source video into FLV.

You can open CPP_Demo.dsw to view the source code for the program.

3. videoposter

We have used ASP to create a simple example: users upload their video files, and then the server script starts the conversion task and pops up a window to show the process of conversion. When the conversion ends, the Flash Video and the link for the Flash Video will be shown on the web page. It needs Persits.Upload component to upload the files, and you can get a trial version from http://www.aspupload.com.

1. Get the source code
   Open the "videoposter" folder in the "Samples" folder under the installation folder for Flash Video MX SDK. The asp source code is under the "videoposter" folder.
2. Make sure you have the IIS installed.
3. Create virtual folder for "videoposter", and make sure to allow anonymous access.
4. Make sure the IIS account IUSR_XXX has full control permission to the **installation** directory. Typically, it's C:\Program Files\Moyea\Flash Video MX SDK
5. Make sure the Moyea Task Dispatcher service is running.
6. Visit "http://127.0.0.1/videoposter/index.html" with browser.

Object Reference

Object Table

The tables below show all the properties and methods of the Object in Moyea Flash Video MX SDK COM.

Encoder Object

Property		Method
Output	EncodingResult	AddAVSource
Width	IsEncoding	AddVideoSource
Height	Thumbnail	AddVideoSourceWithCrop
FPS	ThumbnailWidth	AddAudioSource
KeySpacing	ThumbnailHeight	AddImageWaterMark
VideoBitRate	ThumbnailTime	AddTextWaterMark
StretchMode	ThumbnailFormat	ClearWaterMarks
VideoCodec	ThumbnailStretchMode	ClearSources

SampleRate	currentTime	Cancel
Channels	totalTime	Encode
AudioBitRate	realVideoBtr	EncodeAsTask
VideoFadeInTime	realAudioBtr	Terminate
VideoFadeInColor	SleepTime	WaitForEncodingComplete
VideoFadeOutTime	Priority	
VideoFadeOutColor	UserName	
AudioFadeInTime	Password	
AudioFadeOutTime	Volume	
EncodingMode	Brightness	
Deinterlace	Contrast	
EncodingOKUrl	TaskCompleteCallBack	
EncodingFailedUrl		

EncodingTask Object

Property		Method
TaskID	realVideoBtr	Refresh
State	realAudioBtr	RemoveTask
currentTime	TaskExists	
totalTime	EncodingResult	

VideoInfo Object

Property		Method
HasVideo	AudioChannels	LoadVideo
HasAudio	SampleRate	
Width	Seekable	
Height	UserName	
NeedDeinterlace	Password	
DurationTime	DataRate	
FrameRate		

License Object

Method
SetKeyCode

FLV2SWF Object

Property		Method
FLVPath	LeftBorder	Convert
SWFPath	TopBorder	
Version	RightBorder	
BackgroundColor	BottomBorder	
StopAtStart	VideoWidth	
StopAtEnd	VideoHeight	
Rewind	UserName	
Nosound	Password	

PictureGrabber Object

Property		Method
Source	rightCrop	AddGrabPicture
Width	bottomCrop	AddGrabPictureWithTemplate
Height	PictureFormat	Grab
StretchMode	UserName	
leftCrop	Password	
topCrop		

Encoder Object

All the encoding operation should be implemented with Encoder Object. First, you have to create an "fmx.Encoder" object. Then you should set the properties of the object, such as to set the file to be encoded, the output file, video bit rate, frame rate, size and so on. When finish setting the parameters, you can invoke the "Encode" method, and it starts encoding.

Properties of Encoder Object

Property	Data Type	Description
Output	BSTR	Output file path. Version < 2.0: The output file name will be appended .flv extension. Version 2.0: If the Output has an flv extension, no .flv extension will be appended.
Width	LONG	Width of output file, can only be even number, the default value is 320.
Height	LONG	Height of output file, can only be even number, the default value is 240.
FPS	DOUBLE	Output video frame rate, the default value is 25.
KeySpacing	LONG	Keyframe interval, the default value is 50.
VideoBitRate	LONG	Output video bit rate in bps, the default value is 400000 bps, i.e. 400kbps.
StretchMode	LONG	When the output Width to Height ratio is different from that of the original Width to Height ratio: 0: Stretch the image to full the screen of the output frame (may be distorted) 2: Keep the original ratio of the original Width to Height, and stretch one dimension, padding the other dimension with black color. If the Wi/Hi > Wo/Ho ("i" is the width of the original file, and "o" is the output file), the padding is in vertical dimension. If Wi/Hi < Wo/Ho, the padding is in horizontal dimension. the default value is 0.
VideoCodec	LONG	Video codec, possible value: 0: H263 encoding 1: VP6 encoding The default value is 0, H263 encoding.
SampleRate	LONG	Output audio sample rate, possible value:

		11025, 22050, or 44100, and the default value is 44100.
Channels	LONG	Output audio channel, possible value is: 1 Mono 2 Stereo The default value is 2.
AudioBitRate	LONG	Output audio bit rate, in bps, possible value: 32000, 40000, 48000, 56000, 64000, 80000, 96000, 128000, 160000, 192000, 224000, 256000, 320000.
VideoFadeInTime	LONG	Time for video to fade in, in milliseconds, the default value is 0, i.e. no fade in effect.
VideoFadeInColor	LONG	Video fade in color, in bbggrr format. The default value is 0, i.e. black. For more information, please refer to **Data Type Using** Color section.
VideoFadeOutTime	LONG	Time for video to fade out, in milliseconds, the default value is 0, i.e. no fade out effect.
VideoFadeOutColor	LONG	Video fade out color, in bbggrr format. The default value is 0, i.e. black. For more information, please refer to **Data Type Using** Color section.
AudioFadeInTime	LONG	Time for audio to fade in, in milliseconds, the default value is 0, i.e. no fade in effect.
AudioFadeOutTime	LONG	Time for audio to fade out, in milliseconds, the default value is 0, i.e. no fade out effect.
EncodingMode	LONG	Encoding Mode: 0 Encode both image and audio 1 Encode image only 2 Encode audio only
Deinterlace	LONG	Enable deinterlace or not: 0: Auto (default) 1: Enable
EncodingOKUrl	BSTR	The callback URL when the encoding succeeds in the form of EncodeAsTask. The task ID named “fmx_task_id” will be passed to the URL. **Obsolete in V2.0, use TaskCompleteCallBack instead**
EncodingFailedUrl	BSTR	The callback URL when the encoding fails in the form of EncodeAsTask. The task ID named “fmx_task_id” will be passed to the URL. **Obsolete in V2.0, use TaskCompleteCallBack instead**
EncodingResult	LONG	The encoding result, if the result is 0, it indicates the encoding succeeds; the result with other value indicates the encoding fails.
IsEncoding	BOOL	Check if the encoding is in process. This property is only for the encoding launched by **Encode**

		method. For encoding launched by **EncodeAsTask**, please refer to **EncodingTask** object.
Thumbnail	BSTR	File path for thumbnail picture, its extension determines its format: jpg: JPEG image gif: GIF image png: PNG image bmp: Bitmap If the file path for thumbnail picture is not set, no thumbnail picture will be generated.
ThumbnailWidth	LONG	Width of thumbnail picture, in pixels
ThumbnailHeight	LONG	Height of thumbnail picture, in pixels
ThumbnailTime	LONG	Time to capture thumbnail picture, in milliseconds. The default value is -1, which indicates to capture thumbnail picture in a random time.
ThumbnailFormat	BSTR	Thumbnail format, if the property is empty string, the format will be determined by the thumbnail file extension. Possible value: jpg for JPEG image tiff for TIFF image gif for GIF image bmp for Bitmap image png for PNG image *Available in v2.0.*
ThumbnailStretchMode	LONG	The stretching mothod used when resizing the thumbnail image. Possible value: 0: Direct stretch 1: Create letterbox *Available in v2.0.*
currentTime	LONG	Current time for encoding, in milliseconds. This property is only for the encoding launched by **Encode** method. For encoding launched by **EncodeAsTask**, please refer to **EncodingTask** object.
totalTime	LONG	Total time for encoding, in milliseconds. This property is only for the encoding launched by **Encode** method. For encoding launched by **EncodeAsTask**, please refer to **EncodingTask** object.
realVideoBtr	LONG	Actual video bit rate, in bps. This property is only for the encoding launched by **Encode** method. For encoding launched by **EncodeAsTask**, please refer to **EncodingTask** object.

realAudioBtr	LONG	Actual audio bit rate, in bps. This property is only for the encoding launched by **Encode** method. For encoding launched by **EncodeAsTask**, please refer to **EncodingTask** object.
SleepTime	LONG	Time for the program to stop after encoding one frame, in milliseconds. It is used to prevent the program from increasing CPU usage during the process of encoding. By default, the value is 0, which indicates not to sleep, so that the encoding will be finished as soon as possible.
Priority	LONG	Set the priority of encoding process. 0 normal 1 below normal 2 low
Affinity	LONG	Hexadecimal integer. Runs the process on some set of processors. Binary bit 0 stands for processor 0, while bit 31 stands for processor 31. e.g. 0x5 appears as 101 under binary, and the bit 0 and bit 2 are 1, others are 0, showing that the current process can run on CPU 0 and CPU 2, but can not run on CPU 1 and CPU 3 or above to CPU 31.
UserName	BSTR	The user account used to login for encoding. By default, the current caller's account will be used. For more information, please refer to the **Encoding under another account** section. *Available in v1.5.*
Password	BSTR	The account password used to login for encoding. For more information, please refer to the **Encoding under another account** section. *Available in v1.5.*
Volume	double	Volume adjustment. 1.0 is to keep the original volume, 0.5 to halve the volume, 2.0 to double the volume, etc. *Available in v2.0*
Brightness	LONG	Brightness adjustment. Value is between -127 and 127. -127 is the darkest, and 127 is the brightest. 0 is no change. *Available in v2.0*
Contrast	LONG	Contrast adjustment. Value is between -127 and 127. -127 is the lowest contrast, and 127 is highest contrast. 0 is no change. *Available in v2.0*
TaskCompleteCallBack	BSTR	The callback URL when task is completed no matter if the task is successful or not. A list of variables will be passed to the URL: *task_id*: a string representing the Task ID. *status_code*: an integer value which indicates the task status, -1 is failed, 2 is successful.

		exit_code: the encoding process's exit code, it's the same as EncodingResult of EncodingTask object. *Available in v2.0*

Methods of Encoder Object

Method(**parameters in italics can be default)**	**Description**
AddAVSource(pathVideo, pathAudio, *stTime=0,* *duration=-1*)	Add "pathVideo" file to video track & add "pathAudio" file to audio track. pathVideo: the path for video source file pathAudio: the path for audio source file stTime: video start time, in milliseconds, the default value is "0". duration: video duration, in milliseconds, the default value is "-1", i.e. the original duration of the "pathVideo". **Remark:** Pass empty character string as video path, if you do not want to include video. Pass empty character string as audio path, if you do not want to include audio. Use "pathVideo = pathAudio = source file path" to keep original video and audio in the file. Use "pathVideo=A, pathAudio=B" to replace the audio data in file A with the audio data in file B, and keep original video data in file A.
AddVideoSource(path, *stTime=0,* *duration=-1*)	Add a segment of video source to the target video track. path: the path of the video source stTime: source video start time, in milliseconds, the default value is "0". duration: video duration, in milliseconds, the default value is "-1", i.e. the original duration of the "path". **Remark:** You can use this method to add a segment of source video to the target video track. This method does not like **AddAVSource**, as it doesn't care the audio stream in video file "path". It just splits the video stream from the video file "path" and cuts a segment from the stream by specifying stTime & duration parameters, and then puts the segment to the target video track, which is waiting for encoding. A continuous calling of **AddVideoSource** without calling **ClearSources** will result in that all segments specified by **AddVideoSource** are joined together with the sequence of calling. There is another method named **AddAudioSource** for adding a segment of audio source to the target audio track, the behavior of these two methods are almost the same except that one is dealing with video and the other is dealing with audio. When using these two methods, you can freely arrange any segments of

	video/audio to any position. However, it's your responsibility to keep video & audio in sync.
AddVideoSourceWithCrop(path, *leftCrop=0,* *topCrop=0,* *rightCrop=0,* *bottomCrop=0,* *stTime=0,* *duration=-1*)	Add a segment of cropped video source to the target video track. path: the path of the video source. leftCrop: left cropped pixels, default is 0. topCrop: top cropped pixels, default is 0. rightCrop: right cropped pixels, default is 0. bottomCrop: bottom cropped pixels, default is 0. stTime: source video start time, in milliseconds, the default value is "0". duration: video duration, in milliseconds, the default value is "-1", i.e. the original duration of the "path". Remark: This method is almost the same as AddVideoSource, but allows you to crop the source video before adding to the encoding track. *Available in v2.0*
AddAudioSource(path, *stTime=0,* *duration=-1*)	Add a segment of audio source to the target audio track. path: the path of the audio source stTime: source video start time, in milliseconds, the default value is "0". duration: audio duration, in milliseconds, the default value is "-1", i.e. the original duration of the "path". **Remark:** Please refer to **AddVideoSource**.
AddImageWaterMark(path, *nOrigin=0,* *x=0,* *y=0,* *w=32,* *h=24,* *tranColor=-1,* *torlerance=10,* *alpha=255,* *tOrigin=0,* *tOffset=-1,* *tDuration=-1*)	Add watermark image to video path: File path for image nOrigin: Coordinate origin 0: Top-Left 1: Top-Center 2: Top-Right 3: Center 4: Bottom-Left 5: Bottom-Center 6: Bottom-Right 7: Left-Center *Available in v2.0* 8: Right-Center *Available in v2.0* x, y, w, h: The coordinate, Width and Height of the image in the target video. tranColor: Transparent color in bbggrr format. The default value is -1, which indicates not to be transparent. Please refer to the **Data Type Using** Color section. tolerance: Tolerance for transparent color, possible value ranges from 1 to 255; the default value is 10. alpha: Alpha/Transparency, possible value ranges from 0 to 255; the default value is 255, i.e. no transparency. tOrigin: Time origin, 0 is at the beginning of the video; 1 is in the

	middle of the video; 2 is at the end of the video. tOffset: The time offset compared to time origin, in milliseconds, begin to show watermark at this time point. tDuration: Time duration to show watermark, in milliseconds. Default is "-1", i.e. the watermark will last to the end.
AddTextWaterMark(text, *origin=0,* *x=0,* *y=0,* *c=0xD7A802,* *font= "Times New Roman",* *size=20,* *style=0,* *effect=0,* *alpha=255,* *tOrigin=0,* *tOffset=-1,* *tDuration=-1*)	Add watermark text to video. text: The characters to be shown on the video origin: Coordinate origin 0: Top-Left 1: Top-Center 2: Top-Right 3: Center 4: Bottom-Left 5: Bottom-Center 6: Bottom-Right 7: Left-Center *Available in v2.0* 8: Right-Center *Available in v2.0* x,y: Watermark coordinate compared to origin. c: text color in bbggrr format. Please refer to the **Data Type Using** Color section font: Font name size: Font size style: Font style 0: Normal 1: Italic 2: Bold 3: Bold & Italic effect: Font effect 0: Normal 1: Strikeout 2: Underline alpha: Alpha/Transparency, possible value ranges from 0 to 255; the default value is 255, i.e. no transparency. tOrigin: Time origin, 0 is at the beginning of the video, 1 is in the middle of the video, and 2 is at the end of the video. tOffset: The time offset compared to time origin, in milliseconds, begin to show watermark at this time point. tDuration: Time duration to show watermark, in milliseconds; -1 indicates to show watermark without time limit once it is shown.
ClearWaterMarks	Clear all the added watermarks
ClearSources()	Clear the source file(s) in audio & video encoding track.
Cancel	Notice the current encoding process to cancel encoding, only valid when the encoding process is started in the form of Encode. **Remark:** This method is only to notice the encoding process to cancel encoding, and the encoding process does not stop encoding immediately. You

	need to use **WaitForEncodingComplete** to wait until the encoding process completes. **Terminate** is to compel encoding process to end. When the Terminate method is returned, the encoding process is compelled to end.
Encode()	Start encoding process immediately, and return. **Remark:** When encoding is started, the method will return immediately. If you want it to wait, please read **WaitForEncodingComplete**. **Note:** If you delete an Encoder object that's doing encoding, the WaitForEncodingComplete(-1) will be called automatically.
EncodeAsTask	Add encoding task to encoding queue to implement encoding. The return value is the task ID, through which you can inquire the task info. For more information, please read **EncodingTask Object**. **Note**: If you want to call this method, you have to start Moyea Task Dispatcher service first.
Terminate	Compel encoding process to end, only valid when the encoding process is started in the form of Encode.
WaitForEncodingComplete(tTimeout)	Wait until the encoding process completes; tTimeout is the maximum time to wait until the encoding process completes, in milliseconds; if the encoding process does not complete within tTimeout, it will return False, otherwise it will return True. **Remark:** This method is invalid when the encoding process is started in the form of EncodeAsTask. If tTimeout is -1, it indicates to wait until the encoding process completes. **Note:** If you delete an Encoder object that's doing encoding, the WaitForEncodingComplete(-1) will be called automatically.

EncodingTask Object

EncodingTask object is used to inquire the task state of encoding which is in the form of EncodeAsTask. You can inquire the current encoding state (not encoded, encoding), current time for encoding, total time for encoding, current video bit rate, current audio bit rate and so on. When Encoder add task to encoding task queue with EncodeAsTask, you can use EncodingTask to inquire the task information through task ID. Once the task is finished (fails or succeeds), if you have set TaskCompleteCallBack, TaskCompleteCallBack will be accessed, and then the task will be deleted. Therefore, if you have not set TaskCompleteCallBack, once the task is finished, the task will be deleted, and you can not inquire the task information through EncodingTask object any more. Thus, when using EncodingTask object, you can choose to:

1. Show the encoding process, when the task has been submitted and is waiting for encoding or is encoding.
2. Inquire task information from the script specified by TaskCompleteCallBack.

Note: If you want to use this object, you have to start Moyea Task Dispatcher service first.

Properties of EncodingTask Object

Property	Data Type	Description

TaskID	BSTR	Task ID, Value should be given to TaskID before you inquire certain task information.
State	LONG	Conversion state: -1 Failed 0 Waiting 1 Converting 2 Finished
currentTime	LONG	Current time for conversion, in milliseconds.
totalTime	LONG	Total time for conversion, in milliseconds.
realVideoBtr	LONG	Actual video bit rate, in bps.
realAudioBtr	LONG	Actual audio bit rate, in bps.
TaskExists	BOOL	Check if task exists.
EncodingResult	LONG	Encoding result of the task. *Available in v2.0*

Methods of EncodingTask Object

Method	Description
Refresh	Refresh the statistic data As state, currentTime, totalTime, realVideoBtr, realAudioBtr, TaskExists are not the real time value, you have to use "Refresh" to refresh the current statistic data.
RemoveTask(task_id)	Remove the task specified by the task_id. Only the task in waiting status can be removed. *Available in v2.0*

VideoInfo Object

VideoInfo object is used to get video information, such as image size, frame rate, audio channels, and audio sample rate, and so on.

Properties of VideoInfo Object

Property	Data Type	Description
HasVideo	BOOL	Contain video data or not
HasAudio	BOOL	Contain audio data or not
Width	LONG	Image width, in pixels.
Height	LONG	Image height, in pixels.
NeedDeinterlace	BOOL	Need to deinterlace image or not
DurationTime	LONG	Video duration, in milliseconds
FrameRate	Double	Frame rate; when it can not be detected, it may return "0".
AudioChannels	LONG	Channels
SampleRate	LONG	Sample rate, in Hz.
Seekable	BOOL	Video is seekable or not
UserName	BSTR	The user account used to login for video information detection. By default, the current caller's account will be used. For more information, please refer to the **Encoding under another account** section.

		Available in v1.5
Password	BSTR	The account password used to login for video information detection. For more information, please refer to the **Encoding under another account** section. *Available in v1.5*
DataRate	LONG	Average data rate of the video file in bps. *Available in v2.0*

Methods of VideoInfo Object

Method	Description
LoadVideo(path)	Load the video and get its information. **Return:** "True" for success, "False" for failure. **Remark:** Only when the video has been loaded with LoadVideo, do other properties of this object make sense.

License Object

License object is used to register the product. When you deploy your program, you need to use License object to register Flash Video MX SDK first.

Method of License Object

Method	Description
SetKeyCode(key)	Set registration code key is a character string for registration code **Return:** "True" for success, "False" for failure.

FLV2SWF Object

This COM Object is to convert the FLV file generated by Flash Video MX SDK to SWF file.

Through FLV2SWF Object, and by applying the **Convert** method, you can convert the FLV file into SWF file after FLV2SWF related properties are properly set.

Properties of FLV2SWF Object

Property	Data Type	Description
FLVPath	BSTR	Input the path of FLV.
SWFPath	BSTR	Output the path of SWF.
Version	LONG	Set the version for the output SWF, x can be 7 or 8.
BackgroundColor	LONG	Set the background color in rrggbb format, -1 is for transparent color.
StopAtStart	BOOL	Place a stop action in the first frame.
StopAtEnd	BOOL	Place a stop action in the last frame.
Rewind	BOOL	Place a gotoAndPlay(1) action in the last frame.
Nosound	BOOL	Drop sound stream.

LeftBorder	LONG	Set the width of the left border in pixel.
TopBorder	LONG	Set the height of the top border in pixel
RightBorder	LONG	Set the width of the right border in pixel.
BottomBorder	LONG	Set the height of the bottom in pixel.
VideoWidth	LONG	Set the width of the video in pixel.
VideoHeight	LONG	Set the height of the video in pixel.
UserName	BSTR	The user account is used to login for FLV to SWF conversion. By default, the current caller's account will be used. For more information, please refer to the **Encoding under another account** section.
Password	BSTR	The account password is used to login for FLV to SWF conversion. For more information, please refer to the **Encoding under another account** section.

Methods of FLV2SWF Object

Method	Description
Convert	Convert FLV file to SWF file. **Return:** "True" is for success, and "False" is for failure.

PictureGrabber Object

This COM Object is to grab pictures from the source.

Through PictureGrabber Object, you can set the properties for PictureGrabber. By calling **AddGrabPicture** or **AddGrabPictureWithTemplate**, you can set the parameters for a desired thumbnail picture. Then, you can get thumbnail pictures with the method of **Grab**.

Properties of PictureGrabber Object

Property	Data Type	Description
Source	BSTR	The source to generate images
Width	LONG	The width of the thumbnail picture, in pixel
Height	LONG	The height of thumbnail picture, in pixel
StretchMode	LONG	The stretch mode preferred when resizing the thumbnail images. Possible value: 0: Direct stretch 1: Creat letterbox
leftCrop	LONG	Left cropped pixels of the source
topCrop	LONG	Top cropped pixels of the source
rightCrop	LONG	Right cropped pixels of the source
bottomCrop	LONG	Bottom cropped pixels of the source
PictureFormat	BSTR	The output image format jpg: JPEG image tiff: TIFF image gif: GIF image bmp: Bitmap png: PNG image

UserName	BSTR	User account used to login for image grabbing. By default, the current caller's account will be used. For more information, please refer to the **Encoding under another account** section.
Password	BSTR	The account password used to login for image grabbing. For more information, please refer to the **Encoding under another account** section.

Methods of PictureGrabber Object

<table>
<tr><th>Method</th><th>Description</th></tr>
<tr><td>AddGrabPicture
(filename,
tOrigin=0,
tOffset=-1)</td><td>Generate a thumbnail at a specified time.
Generate thumbnail pictures at a specified time.
Filename: the path of the thumbnail generation.
tOrigin：time origin of the thumbnail grabbing. 0: video start time; 1: video middle time; 2: video end time。
tOffset：offset value from the time origin. Positive value indicates rightward from the time origin, while negative indicates leftward from the time origin.
The logic between the tOrigin and tOffset is
<table>
<tr><td>tOrigin</td><td>tOffset</td></tr>
<tr><td>0</td><td>0 or positive values</td></tr>
<tr><td>1</td><td>any values</td></tr>
<tr><td>2</td><td>0 or negative values</td></tr>
</table>
This method can be applied many times to generate series of thumbnail pictures.</td></tr>
<tr><td>AddGrabPictureWithTemplate
(number,
filename_template)</td><td>Grab numbers of images and save with the name as specified by filename_template. The filename_template string must contain “?”, which will be replaced by the number when the pictures are generated.
For example, set the number as 5, then the filename_template is C:\demo??.bmp, and the image series will be
C:\demo01.bmp, C:\demo02.bmp,…, C:\demo05.bmp.
This method can be applied many times for series of thumbnails to be generated.</td></tr>
<tr><td>Grab</td><td>Generate thumbnail pictures.
Return:
“True” is for success, and “False” is for failure.</td></tr>
</table>