Grimm | AUDIO

LevelShow

Please read this manual before operating the software!

Manual written by Jorn T. Lemon

Software designed and developed by Jorn T. Lemon and Wouter D. Snel
And thanks to Stijn van Beek

Introduction

Thank you very much for selecting Grimm Audio's LevelShow for your audio normalizing tasks. This program is build to support one of the most important achievements in the audio industry for decades: the change from peak normalization to loudness normalization. This "true audio revolution" started when the ITU submitted the BS.1770 'LKFS' (also known as LUFS) loudness metering standard in 2006. The European Broadcast Union EBU took the lead in building a broadcast recommendation upon this fundament, called R128. It was released in 2010. Eelco Grimm of Grimm Audio was one of the active members of the EBU PLOUD committee that created the recommendation. The committees work has been made possible by the aid of a piece of software developed by Grimm Audio's Wouter Snel and Jorn Lemon. This means LevelShow is the enhanced version of the actual software that made R128 happen. We are very proud to offer you this great piece of software that will enhance your daily work spectacularly.

System Requirements

Minimum System Requirements:

- XP , Vista or Windows 7 (x86 or x64)
- System Memory 512 mb
- Available disk space: approximately 10mb

Installation

After downloading the setup file, double-click it to start the installer. You might see a general security warning that the file is downloaded from a website. If so, just press the "run" button. The installer will guide you through the install process. You don't need to change any of the settings unless you wish. When finished you may remove the setup file. In the start menu you will find a LevelShow folder that contains a a link to the header folder (needed for programmatic control), the LevelShow-Registration and the uninstaller.

In the start menu you will find a LevelShow folder that contains a a link to the header folder (needed for programmatic control), the LevelShow-Registration and the uninstaller.

Please note: the LevelShow registration wizard requires .NET to be installed. If you do not wish to install .NET you can install the "Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)". Which is a lot smaller (4mb) but installs everything needed to run LevelOne.

Registration

Launch the registration wizard from the start menu under LevelShow. This wizard will guide you through the registration process. If you have had a previous version of LevelShow install on the system the wizard will automatically adopt the settings of the previous installation.

Please note: the LevelShow requires the Microsoft Visual C++ 2010 SP1 Redistributable Package to be installed. this will be done automatically during the installation.

Usage

The following steps will describe the basic usage of the LevelShow plugin.

1. Create a valid DirectShow graph that connects to the in and output pins of a LevelShow instance.
2. Make sure all the Analyse settings are correct. (open the filter properties page to open the graphical interface, or set the following instructions programmatically)
    1. Mode should be set to Analyse.
    2. Choose the desired peak measurement type.
    3. In case of a 6 channel surround stream. Select the correct surround channel order.
    4. Set the LU calibration if you need to. The initial default is set to -23 LUFS (according to EBU R-128)
    5. Set the PPM calibration if you need to. The initial default is set to -9 dBFS.
    6. Turn the LU Relative gate on or off. Initially it is on (according to EBU R-128)
    7. Choose the desired target type and level. Default is LU with a level of 0 (according to EBU R-128)
3. After setting up the properties you can start to stream the audio to LevelShow, in real time or faster.
4. Depending on the implementation, (see the "Implementation" paragraph below) when all audio data has been streamed you can now:
5. with 1 instance, set the mode property to Adjust, which will automatically set LevelShow to the correct adjustment level.
6. with 2 instances, readout the adjustment level and save it. To reinitialize LevelShow just switch to Analyse and back to Adjust (or programmatically set mode to -1. for a quick reset)
7. Lastly you can either:
    1. Restream the audio data through the same instance of LevelShow to adjust the audio to the target level.
    2. Use the saved adjustment level to set the adjust level of another LevelShow instance that is set to Adjust mode. And restream the audio though it to adjust it to the target level.

Implementation

Implementing the LevelShow directshow filter to adjust an audio stream can be achieved in 2 ways:

1. By using 1 instance of the filter in your graph for analyzing and adjusting, which is useful for a quick workflow.
2. Or use 2 instances of the filter to separate the analyze and adjustment tasks. Which will allow you to first analyse multiple files, saving the results to adjust the same files at a later time if needed.

Controlling

Controlling the plugin can be achieved in 2 ways:

1. The graphical user interface (see the "Graphical interface" chapter)
2. The programmatic interface ILevelShow, using the 2 instances implementation method it will be best to implement this programmatically (see the "Programmatic interface" chapter)

Graphical interface

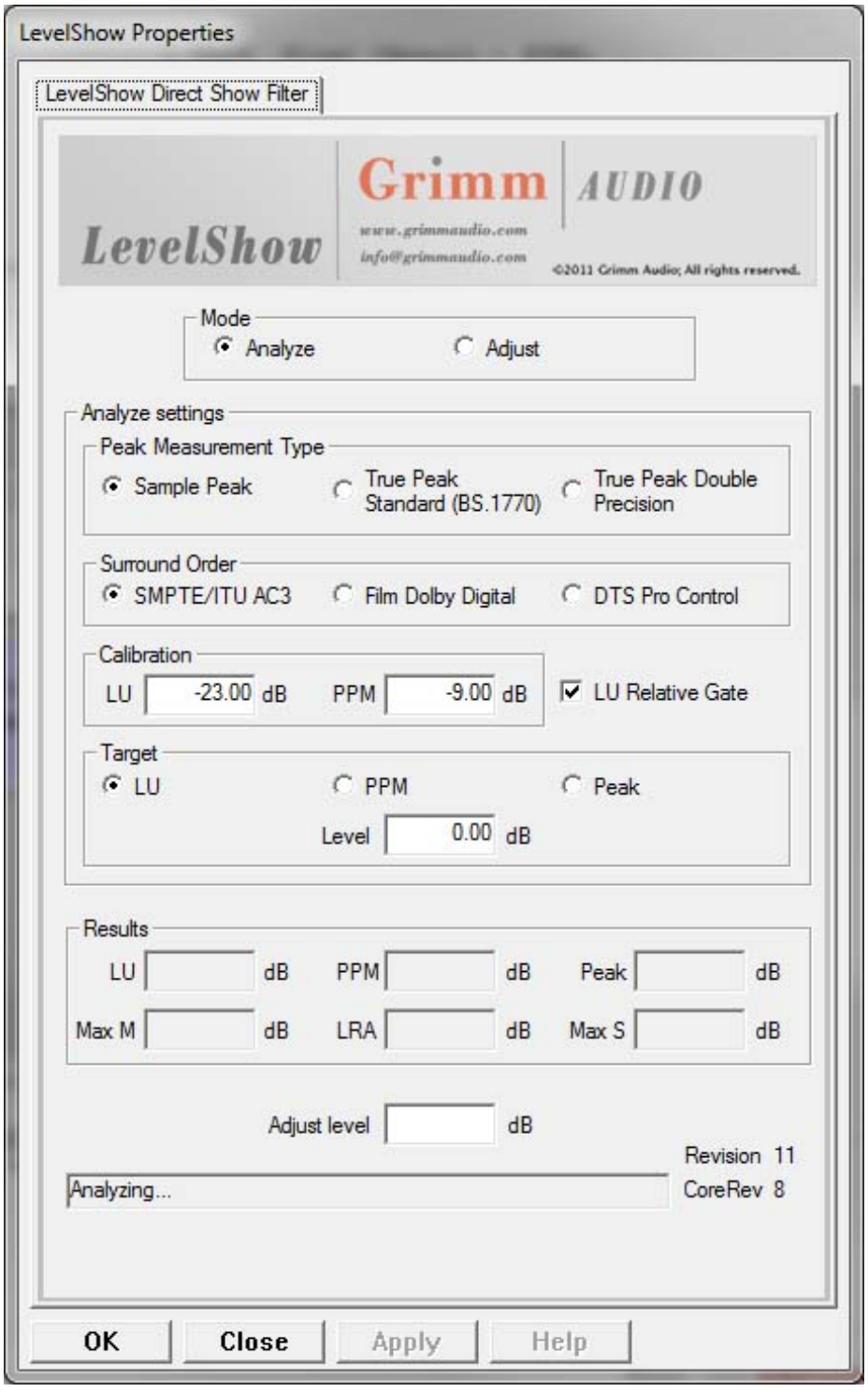
When opening the properties page in for example Graphedit you will see the interface which corresponds to the programmatic interface.

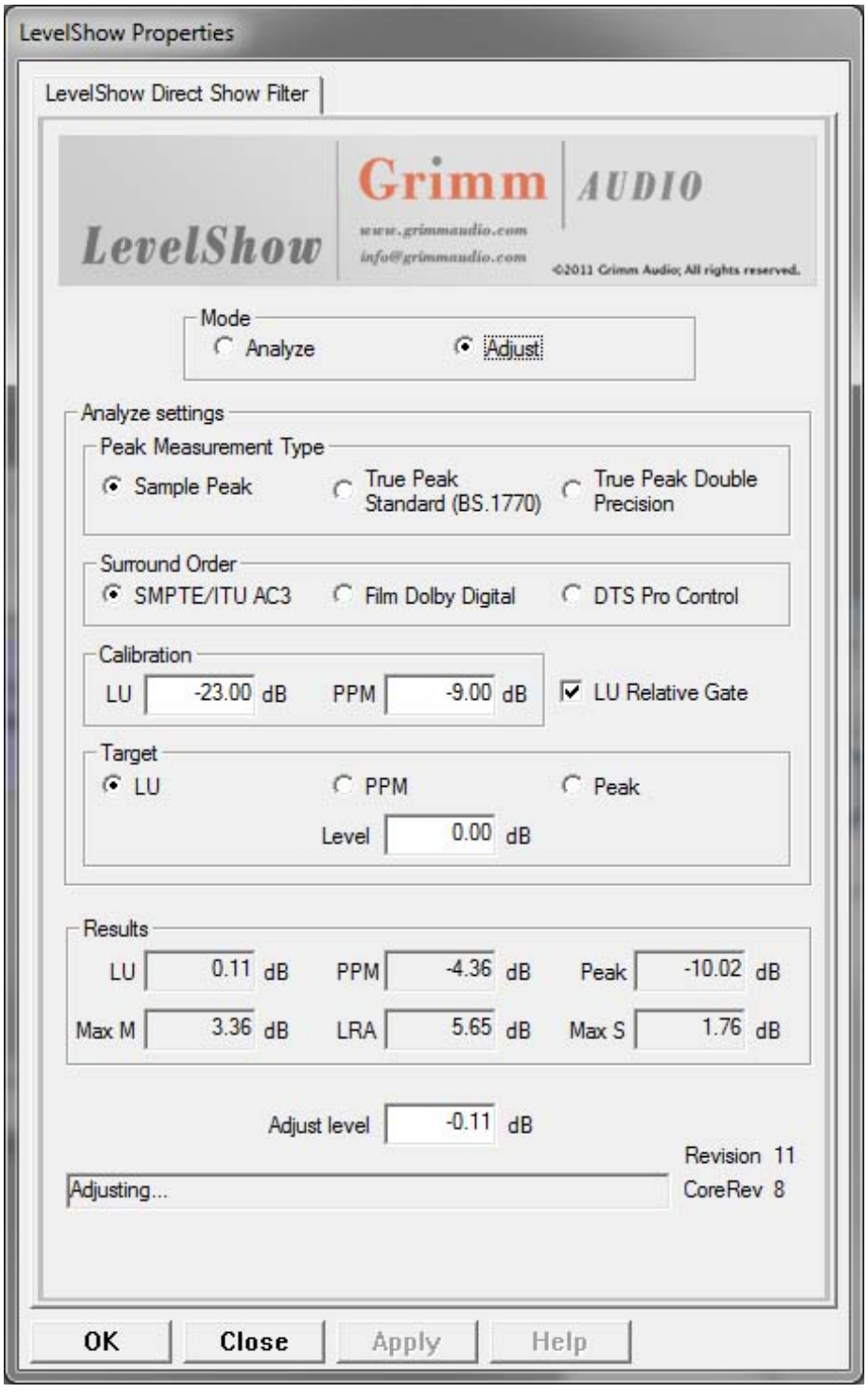
It consists of 3 sections:

Mode
This sets the main function for LevelShow to analyse an audio signal or adjust audio to a measured or a manually set adjustment level.

Analyze settings
These settings will determine the results and adjustment level. See “Usage” chapter for more information.

Results
When all audio is measured and the results are needed, switch the mode to Adjust. Then the results and the adjust level will be displayed. And LevelShow will be automatically set up to adjust the incoming stream.


LevelShow Properties
LevelShow Direct Show Filter
LevelShow
Grimm AUDIO
www.grimmaudio.com
info@grimmaudio.com
©2011 Grimm Audio; All rights reserved.
Mode
Analyze
Adjust
Analyze settings
Peak Measurement Type
Sample Peak
True Peak Standard (BS.1770)
True Peak Double Precision
Surround Order
SMPTE/ITU AC3
Film Dolby Digital
DTS Pro Control
Calibration
LU -23.00 dB
PPM -9.00 dB
LU Relative Gate
Target
LU
PPM
Peak
Level 0.00 dB
Results
LU dB
PPM dB
Peak dB
Max M dB
LRA dB
Max S dB
Adjust level dB
Revision 11
Analyzing...
CoreRev 8
OK
Close
Apply
Help


LevelShow Properties
LevelShow Direct Show Filter
LevelShow
Grimm AUDIO
www.grimmaudio.com
info@grimmaudio.com
©2011 Grimm Audio; All rights reserved.
Mode
Analyze
Adjust
Analyze settings
Peak Measurement Type
Sample Peak
True Peak Standard (BS.1770)
True Peak Double Precision
Surround Order
SMPTE/ITU AC3
Film Dolby Digital
DTS Pro Control
Calibration
LU -23.00 dB
PPM -9.00 dB
LU Relative Gate
Target
LU
PPM
Peak
Level 0.00 dB
Results
LU 0.11 dB
PPM -4.36 dB
Peak -10.02 dB
Max M 3.36 dB
LRA 5.65 dB
Max S 1.76 dB
Adjust level -0.11 dB
Revision 11
CoreRev 8
Adjusting...
OK
Close
Apply
Help

Programmatic interface

The programmatic interface is based on IBaseFilter for the usual directshow connections. For the plugin specific settings you can query the exposed ILevelShow interface.

For more information about interface query's see "IUnknown::QueryInterface Method": http://msdn.microsoft.com/en-us/library/ms682521%28VS.85%29.aspx.

ILevelShow is inherited from the IUnknown interface and defines the following methods: (also see 'Appendix 1 - LevelShow Header')

Stream info

Standard stream info

```
ILevelShow::get_BytesPerSample int
ILevelShow::get_Channels int
```

Software info

License check Return Value: false=Invalid license, true=Valid license

```
ILevelShow::get_License bool
```

Revision check
Return Value: an integer from 0 and up, indicates the revision of the software core.

```
ILevelShow::get_Revision int
```

Main Setting

Mode
put Parameter: 0=analyze 1=adjust or -1=Reset, to analyse a new stream and forget previous measurements
get Return Value: 0=analyze 1=adjust

```
ILevelShow::get_Mode int
ILevelShow::put_Mode int
```

Analyse settings

Peak Measurement Type

put Parameter: 0=Sample Peak, 1=True Peak Standard (BS.1770) 2=True Peak Double Precision (default is 0)

get Return Value: 0=Sample Peak, 1=True Peak Standard (BS.1770) 2=True Peak Double Precision

```
ILevelShow::get_PeakMeasurementType int
ILevelShow::put_PeakMeasurementType int
```

LU Relative Gate

put Parameter: false=off, true=on (default is true)

get Return Value: false=off, true=on

```
ILevelShow::get_LURelativeGate bool
ILevelShow::put_LURelativeGate bool
```

Surround Order

put Parameter: 0=SMPTE/ITU AC3, 1=Film Dolby Digital, 2=DTS Pro Control (default is 0)

get Return Value: 0=SMPTE/ITU AC3, 1=Film Dolby Digital, 2=DTS Pro Control

```
ILevelShow::get_SurroundOrder int
ILevelShow::put_SurroundOrder int
```

Target Type

put Parameter: 0=LU, 1=PPM, 2=Peak (default is 0)

get Return Value: 0=LU, 1=PPM, 2=Peak

```
ILevelShow::get_MeterType int
ILevelShow::put_MeterType int
```

Calibration LU

put Parameter: floating point from -99. to 99. (default is -23.00 for R-128)

get Return Value: floating point from -99. to 99.

```
ILevelShow::get_CalibrationLU float
ILevelShow::put_CalibrationLU float
```

Calibration PPM in dB

put Parameter: floating point from -99. to 99. (default is -9.00)
get Return Value: floating point from -99. to 99.

```
ILevelShow::get_CalibrationPPM float
ILevelShow::put_CalibrationPPM float
```

Target Level in dB

put Parameter: floating point from -99. to 99. (default is 0.00)
get Return Value: floating point from -99. to 99.

```
ILevelShow::get_TargetLevel float
ILevelShow::put_TargetLevel float
```

Adjust Level

An overwrite function that allows you to adjust an audio signal without analyzing the signal in the same filter instance
put Parameter: floating point from -99. to 99.

```
ILevelShow::put_AdjustLevel float
```

Results

Levels

The functions below should be called after all the audio data has passed trough the filter. It can be called after each block of processed samples but
get Return Value: floating point from -99. to 99. (the respective value in dB)

```
ILevelShow::get_LULevel float
ILevelShow::get_PPMLevel float
ILevelShow::get_PeakLevel float
ILevelShow::get_MaxMLevel float
ILevelShow::get_LRALevel float
ILevelShow::get_MaxSLevel float
```

Adjust Level

The calculated adjustment in dB. Useful when combined with 'put_AdjustLevel' in another instance of LevelShow

```
ILevelShow::get_AdjustLevel float
```

Warnings

Warning string in case of an abnormal adjust value it will be corrected. To find out why; supply an string of 256 chars to this function. It will return a zero terminated string (maximum 256 long). If there is no warning it will return a pointer to NULL.
Current possible warnings are:

- Invalid target value, adjust has been reset to 0 dB (short content?)
- Adjust level exceeds %.1f dB it has been lowered as precaution!
- Adjust level exceeds %.1f dB it has been limited as precaution!
- Adjust level has been lowered as precaution to prevent 0 dBFS clipping!

  ILevelShow::get_Warning char

Realtime Levels

For realtime meter indication use the functions below for Momentary and Short-term LU values. These functions can be called after each processed block of samples.

```
ILevelShow::get_MLevel float
ILevelShow::get_SLevel float
```

References

http://tech.ebu.ch/loudness provides extensive information about the EBU R128 broadcast loudness recommendation. The official R128 documents and guidelines can be found, as well as introduction papers and videos.

Grimm Audio CV
Strijpsestraat 94
5616 GS Eindhoven
The Netherlands

Phone: +31 (0)40-213 1562

Email: info@grimmaudio.com
Website: http://www.grimmaudio.com

Appendix 1 - LevelShow Header

```
// ILevelShow's GUID

DEFINE_GUID(IID_ILevelShow, 0x738f22a0, 0x93f9, 0x4247, 0xb9, 0x4c, 0x78, 0x69, 0xce, 0x68, 0x70, 0x17);


// ILevelShow macros to control the filter and readout the results

DECLARE_INTERFACE_(ILevelShow, IUnknown)
{
    // Stream info
    STDMETHOD( get_BytesPerSample ) ( THIS_ int *BytesPerSample ) PURE;
    STDMETHOD( get_Channels )       ( THIS_ int *Channels ) PURE;

    // Core Version/Revision check
    STDMETHOD( get_Revision )       ( THIS_ int *Revision ) PURE;
    // License check { false=Invalid license, true=Valid license}
    STDMETHOD( get_License )        ( THIS_ bool *Version ) PURE;

    // mode (0=analyze 1=adjust)
    // or (-1=Reset, to analyse a new stream and forget previous measurements)
    STDMETHOD( get_Mode )           ( THIS_ int *Mode ) PURE;
    STDMETHOD( put_Mode )           ( THIS_ int Mode ) PURE;

    // Analyse settings
    // PeakMeasurementType
    // { 0=Sample Peak, 1=True Peak Standard (BS.1770) 2=True Peak Double Precision }
    // default is 0
    STDMETHOD( get_PeakMeasurementType )    ( THIS_ int *PeakMeasurementType ) PURE;
    STDMETHOD( put_PeakMeasurementType )    ( THIS_ int PeakMeasurementType ) PURE;

    // LURelativeGate
    // { false=off, true=on } default is true
    STDMETHOD( get_LURelativeGate ) ( THIS_ bool *LURelativeGate ) PURE;
    STDMETHOD( put_LURelativeGate ) ( THIS_ bool LURelativeGate ) PURE;
    // SurroundOrder
    // { 0=SMPTE/ITU AC3, 1=Film Dolby Digital, 2=DTS Pro Control } default is 0
    STDMETHOD( get_SurroundOrder )  ( THIS_ int *SurroundOrder ) PURE;
    STDMETHOD( put_SurroundOrder )  ( THIS_ int SurroundOrder ) PURE;
    // TargetType
    // { 0=LU, 1=PPM, 2=Peak } default is 0
    STDMETHOD( get_MeterType )      ( THIS_ int *MeterType ) PURE;
    STDMETHOD( put_MeterType )      ( THIS_ int MeterType ) PURE;

    // CalibrationLU
    // {-99. to 99. } in dB, default is -23.00 for R-128
    STDMETHOD( get_CalibrationLU )  ( THIS_ float *CalibrationLU ) PURE;
    STDMETHOD( put_CalibrationLU )  ( THIS_ float CalibrationLU ) PURE;
    // CalibrationPPM
    // {-99. to 99. } in dB, default is -9.00
    STDMETHOD( get_CalibrationPPM ) ( THIS_ float *CalibrationPPM ) PURE;
    STDMETHOD( put_CalibrationPPM ) ( THIS_ float CalibrationPPM ) PURE;
    // TargetLevel
    // {-99. to 99. } in dB, default is 0.00
    STDMETHOD( get_TargetLevel )        ( THIS_ float *TargetLevel ) PURE;
    STDMETHOD( put_TargetLevel )        ( THIS_ float TargetLevel ) PURE;

    // An overwrite function that allows you to adjust an audio signal
    // without analyzing the signal in the same filter instance
    STDMETHOD( put_AdjustLevel )        ( THIS_ float AdjustOverwrite ) PURE;

    // The 'result' funtions below must be called after
    // all the sampledata has passed trough the filter. Results are in dB
    STDMETHOD( get_LULevel )        ( THIS_ float *Result ) PURE;
    STDMETHOD( get_PPMLevel )       ( THIS_ float *Result ) PURE;
    STDMETHOD( get_PeakLevel )      ( THIS_ float *Result ) PURE;
    STDMETHOD( get_MaxMLevel )      ( THIS_ float *Result ) PURE;
    STDMETHOD( get_LRALevel )       ( THIS_ float *Result ) PURE;
    STDMETHOD( get_MaxSLevel )      ( THIS_ float *Result ) PURE;

    // These realtime functions can be called after each processed block of samples
    STDMETHOD( get_MLevel )     ( THIS_ float *Result ) PURE;
    STDMETHOD( get_SLevel )     ( THIS_ float *Result ) PURE;

    // The calculated adjustment in dB. Can be used with 'put_AdjustLevel'
    // in another instance of LevelShow
    STDMETHOD( get_AdjustLevel )        ( THIS_ float *Result ) PURE;

    // In case of an abnormal adjust value it may be corrected.
    // To check and find out why; supply an pointer to a string of 256 chars as an argument.
    // It will return a zero terminated string (maximum 256 long)
    // If there is no warning it will return a pointer to NULL
    STDMETHOD( get_Warning )        ( THIS_ char *Warning ) PURE;
};
```

Appendix - Change Log

rev 33+

- Fix for in preset stream, now includes the mode setting
- Added deauthorization license mechanism

rev 27+

- 64 bit support
- Support for realtime Momentary and Short term Loundness values