Windows VisionableSDK v1.3.6 Release Notes (January 8, 2025)

CHANGES/FIXES

Update logging to open file once and flush content after full log message has been written to filestream

V1.3.5 RELEASE NOTES:
CHANGES/FIXES

Updated Audio and Video library dependencies
Crash fix during timezone reading on Windows 10 1809
Corrected Network Stats types and added streamId
Added configuration capability to Audio and Video

V1.3.3 RELEASE NOTES:
CHANGES/FIXES

Improvements to WebSocket handling
Fix for local PTZ camera control
Fix for device info updates during join to meeting
Relative zoom instead of absolute for CAM520

V1.3.2 RELEASE NOTES:
API CHANGES

Added new callbacks for audio and video network conditions.

```
public void audioConditionUpdate(const AudioStatistic& ac);
public void videoConditionUpdate(const VideoStatistics& vc);
```

Added dedicated objects for conditions data:

`AudioStatistic –` represents general audio network condition and contains data for audio streams
`AudioStreamStatistic -` represents audio stream specific network conditions
`VideoStatistic -` represents general video network condition and contains data for video streams
`VideoStreamStatistic -` represents video stream specific network conditions

CHANGES/FIXES

Updated packaging of Meeting SDK, now the folder structure is changed to:
bin – Dynamic libraries with PDBs for CoreMeeting and MeetingSDK.
include – include files needed for library
lib – Link library for compile time linkage
models – tflite models for SDK usage

Updates to Visionable types, that are used instead of regular STL objects. All array types are now using template type: **ObjectsArray<T>** for common manipulations with array data. Only **WindowInfo**, **VideoStreamStatistics** and **AudioStreamStatistic** are available for use.

KNOWN ISSUES

None

V1.3.1 RELEASE NOTES:
CHANGES/FIXES

Minor fixes to ImageCapture/FileCaptureDevice functionality introduced in v1.3 release

Retrofit all SDK APIs to use standard “C” data types instead of C++/STL types. Precise documentation of these changes will be provided in a future set of release notes.

V1.3.0 RELEASE NOTES:
API CHANGES

Added APIs to allow for an "Image Capture" device. This is a device that the application "creates" with an API call by specifying a directory to which image files can be written to (via a new API call) and from which the underlying video engine can read image files to be sent up into a meeting.

```
int enableImageCaptureDevice(std::string& displayName,
          std::string& directory, std::string& mode)
```

Asks the SDK to create a new image device. The `displayName` parameter is the name that will appear in the corresponding `VideoInfo siteName` field for this stream. The `directory` parameter is an absolute path to a directory on the local device that can be written to. This absolute path **must** contain a trailing directory separator. The `mode` parameter is a screen sharing mode to be used for this stream (such as "BEST SCREEN").

Returns an integer ID to be used with other API calls that need to reference this device. Successful execution of this API call will generate an immediate `participantVideoAdded` callback for this user in all applications connected to the meeting.

```
bool disableImageCaptureDevice(int deviceId);
```

Disables a previously created image capture device. The `deviceId` parameter is the identifier returned by the corresponding call to `enableImageCaptureDevice` (which created this capture device). Returns a `boolean` indicating whether or not the call was successful.

```
bool imageCaptureDevicePutImage(int deviceId,
     const uint8_t *yuv420p_ptr, int width, int height, int size);
```

Send a YUV420P image into the meeting for the specified device. The `deviceId` parameter is the identifier returned by the call to `enableImageCaptureDevice` the application used to create the capture device being used. The `yuv420p_ptr` parameter is a pointer to a memory block containing an unpadded YUV420P image. The `width` and `height` parameters are the width and height of the image, respectively. The `size` parameter is the size of the memory block being passed in.

Returns a Boolean indicating whether or not the image was successfully received. This call will immediately write the data in the memory block passed to a file in the directory specified when creating the corresponding capture device. Once the call is complete, you are free to delete the memory block.

CHANGES/FIXES

Miscellaneous improvements to audio/video engine

KNOWN ISSUES

None