Media Recorder API

Warning: This API is not part of the public SDK. It can be found in the SDK API Plug-in. NOTE that these APIs are not supported in any way on Symbian^3 or later!

API Purpose

The media recorder captures and encodes video and audio data using services provided by other subsystems belonging to the Camcorder plug-in, MMF, and platform. The media recorder is implemented as a separate DLL, and can be re-used as a stand-alone component for recording video and audio. The recorder module provides a VCR –type (video cassette recorder) interface with normal initialization, transport control and de-initialization functionality. Media recorder itself is not a MMF plug-in. The main reasons are that it controls both audio and video simultaneously, and that there is no direct support for video processing in the MMF.

Use cases

The media recorder provides an audio-video recording service. The interface includes appropriate initialization functionality (codecs, frame rate, bit rate etc.) and VCR –type functions for controlling the recording.

Supported Format:
Camcorder MMF Plugin provides support for the following formats

Audio coding formats:

- AMR-NB audio (identified with FourCC “ AMR”).
- QCELP audio (identified with FourCC “ Q13”).
- AAC-LC audio at 8 or 16 kHz and mono (identified with FourCC “ AAC”).

Video coding formats

- H.263 profile 0 video, levels 10, 20, and 45.
- MPEG-4 Visual Simple Profile video, levels 0 to 3 (MIME-types “video/mp4v-es; profile-level-id=x”, where x can be 1, 2, 3, 8 or 9; id’s per MPEG-4 standard).

File formats:

- 3GPP multimedia file format (3GP or MIME-type “video/3gpp”).
- 3GPP2 multimedia file format (3G2 or MIME-type “video/3gpp2”)
- MP4 file format (MIME-type “video/mp4”)

The availability of codec plugins in the phone may restrict the actual list of supported formats.

User guide

The user of the media recorder API should call each and every Set function there is in the API before preparing it for recording. This ensures that all recording parameters are suitable for the application in question. The recorder does use reasonable default values for parameters that have not been set by the user, but it is better to specify them explicitly.

The desired video frame size must be set (using `CCMRVideoRecorder::SetVideoFrameSize()`) before setting the desired video frame rate (using `CCMRVideoRecorder::SetVideoFrame()`).

The recorder will return an error if frame size has not been set when setting the frame rate. This is because the selected frame size determines the maximum available frame rate that the camera HW can quarantee. Note that the camera may be asked to give a higher frame rate than the requested coding frame rate, if it does not support the coding frame rate. If frame size is changed later, frame rate is adjusted accordingly to meet camera’s capabilities.

Example code

```
class CmyOwnSink : public Cbase, public MCMRMediaSink
```

```
{
…
}
class CmyClass: public Cbase, public MCMRMediaRecorderObserver
{
…
}

void CmyClass::Init()
{
     iMediaRecorder = CCMRMediaRecorder::NewL();

     iMediaRecorder->GetSupportedVideoCodecsL( dataTypes );
     //decide which to use

     iMediaRecorder->OpenL(this, iAudioSource, iMySink,
                           iCameraHandle, *iVideoCodec, iAudioCodec );
     //iMySink is the dummy sink object
     iMediaRecorder->SetAudioPriorityL( iPrioritySettings );

     iMediaRecorder->SetVideoFrameSizeL(iFrameSize);
     iMediaRecorder->SetVideoFrameRateL(iFramesPerSecond);
     iMediaRecorder->SetAudioEnabledL(aEnable);
     //etc
}

void CmyClass::Run()
{
     iMediaRecorder->RecordL();
     //do something

     iMediaRecorder->PauseL();
     //do something

     iMediaRecorder->ResumeL();
     //do something

     //this can be set “any time”
     iMediaRecorder->SetVideoBitRateL();
     //do something
     iMediaRecorder->StopL();

     delete iMediaRecorder;
}
```

Example project

The following project demonstrates the use of Camcorder Media Recorder API. It has been tested successfully on Nokia N80 and Nokia N95. The application reserves the camera and records 10 seconds video in H.263 format with AMR-NB audio and displays some statistics of the recorded data (number of recorded video + audio frames, average bitrates etc.) Apart from that,audio frames are saved to a file by adding AMR 6 byte header so that the file can be used by other applications as well.similarly the recorded video frames can be saved to file To record using other video formats (MPEG-4) and higher resolutions, information from the following solution could be applied to this API as well:	File:CCMRTest.zip

TSS000680 - Getting a larger size video recorded by CVideoRecorderUtility