Archived:Audio Routing API – Input and Output

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Audio Output Routing

Note: :This API is not part of the public SDK. It can be found in the SDK API Plug-in.

The Audio Routing API provides way for controlling audio routing in 3rd Edition Feature Pack 1 & 3rd Edition Feature Pack 2 devices; the audio can be routed to earpiece or loudspeaker or both.

Use cases

To route audio from loudspeaker to earpiece/headset.

To route audio from earpiece to loudspeaker(even when headset is connected).

Example code

Header files

```
#include <AudioOutput.h>           // for audio routing control
#include <MAudioOutputObserver.h>  // for audio routing observers
#include <MdaAudioSamplePlayer.h>  // for playing audio
```

Libraries Used

```
audiooutputrouting.lib    // for routing audio
mediaclientaudio.lib      // for playing audio
```

When we play music files using CMdaAudioPlayerUtility class, the audio will be routed to loudspeaker by default. When the headsets are connected, the audio is routed from loudspeaker to headsets by default.

The CAudioOutput class in AudioOutput.h is used by an audio application to inform the Audio subsystem on where the audio need to be routed for output when playing.This class should only be used if the default audio routing is not sufficient for the audio client.

The following example code shows how to control audio routing:

Play audio file with a CMdaAudioPlayerUtility instance. Playing audio files using CMdaAudioPlayerUtility is explained Playing audio files using Symbian C++.

The audio is played to loudspeaker, by default if headsets are not connected.Once the audio file playing is started, then you can create the CAudioOutput instance.(by passing the playerutility to it)

RegisterObserverL() allows clients to register to be updated when the default audio output changes and later when the specified observer no longer wants to be updated call UnregisterObserver().

Then call SetAudioOutputL() passing EPrivate as TAudioOutputPreference parameter for the SetAudioOutputL() API for routing audio to earpiece.

```
// To route audio to private speaker(earpiece)
CAudioOutput::TAudioOutputPreference myOutputPref = CAudioOutput::EPrivate;
iMySound->SetRoutingL(myOutputPref);

void CMySound::SetRoutingL(CAudioOutput::TAudioOutputPreference& aAudioOutput)
 {
 iAudioOutput = CAudioOutput::NewL(*iMyAudioPlayerUtility);
 iAudioOutput->RegisterObserverL(*this);
 iAudioOutput->SetAudioOutputL(aAudioOutput);
 }
```

```
// callback function
void CMySound::DefaultAudioOutputChanged( CAudioOutput& aAudioOutput,
                    CAudioOutput::TAudioOutputPreference NewDefault )
       {
       // Audio routing changed, write your code here
       CEikonEnv::InfoWinL(_L("In Callback function"),_L("AudioOutput Routed"));
       }
```

```
// To route audio to public speaker(loudspeaker),even when headset is connected
CAudioOutput::TAudioOutputPreference myOutputPref = CAudioOutput::EPublic;
```

```
iMySound->SetRoutingL(myOutputPref);
```

```
// To route audio to both speakers(loudspeaker & earpiece)
CAudioOutput::TAudioOutputPreference myOutputPref = CAudioOutput::EAll;
iMySound->SetRoutingL(myOutputPref);
```

Following are the various TAudioOutputPreference enums defined in AudioOutput,h, which can be passed to SetAudioPutputL() API :

The output audio can be routed as desired by choosing proper TAudioOutputPreference parameter(as per your routing option) for the SetAudioOutputL() API.

```
enum TAudioOutputPreference
   {
   ENoPreference, /// Used to indicate that the playing audio can be routed to
                  /// any speaker. This is the default value for audio.

   EAll,          /// Used to indicate that the playing audio should be routed
                  /// to all speakers.

   ENoOutput,     /// Used to indicate that the playing audio should not be routed
                  /// to any output.

   EPrivate,      /// Used to indicate that the playing audio should be routed to
                  /// the default private speaker. A private speaker is one that
                  /// can only be heard by one person.

   EPublic        /// Used to indicate that the playing audio should be routed to
                  /// the default public speaker. A public speaker is one that can
                  /// be heard by multiple people.
   };
```

Example project

File:Control Audio Routing.zip

Audio Input Routing

CS001029 - Audio Input Routing API