Archived:MaoscPlayCompete() callback not received after output stream underflow (Known Issue)

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*}}.

When using `CMdaAudioOutputStream`, `MaoscPlayCompete()` callback is not received automatically after all the buffers have been sent to server side and played back.

Description

`CMdaAudioOutputStream` uses callbacks (`MMdaAudioOutputStreamCallback`) to notify the client of the playback progress. According to SDK documentation, `MaoscPlayComplete()` callback is received when all descriptors have been sent.

However, `MaoscPlayComplete()` callback is received only after an explicit call to `CMdaAudioOutputStream::Stop()`, not after the all the buffers have been processed and the output stream has underflowed.

Solution

After the output stream underflows, it will continue to wait for new buffers. At this point, the only way of releasing the stream is to call `CMdaAudioOutputStream::Stop()`.

Since no notification about the underflow is received, the client will need some mechanism to determine when to stop the stream. Specifically, the client needs to know which buffer will be the last one to be played back. When (a pointer to) the last buffer is known, it is possible to detect when it has been copied to server side in `MaoscBufferCopied`, and then proceed to stop the stream.

On S60 2nd Edition devices, a call to `CMdaAudioOutputStream::Stop()` should not be made directly inside `MaoscBufferCopied()`, as it may result in a panic. Instead, the client can start a low-priority active object (`CIdle`), which will then call `Stop()` once it completes.

The following code demonstrates stopping the output stream with an active object:

```
void CAudioStreamEngine::ConstructL()
    {
    ...
    // Construct the active object (CIdle) used for stopping the stream
    iStop = CIdle::NewL( EPriorityIdle );
    }

void CAudioStreamEngine::MaoscBufferCopied( TInt aError, const TDesC8& aBuffer )
    {
    if( aError == KErrNone )
        {
        // Compare the copied buffer to the known last buffer
        if( &aBuffer == iLastBuffer )
            {
            // Playback is complete:
            // Start the active object that will stop the stream
            iStop->Start( TCallBack(BackgroundStop, this) );
            return;
            }
        // Write the next playback buffer to the stream
        else
            {
            iOutputStream->WriteL( GetNextBuffer() );
            }
        }
    else
        {
        // Error handling
        }
    }

TInt CAudioStreamEngine::BackgroundStop( TAny *aStream )  // static member function
    {
    return ((CAudioStreamEngine*)aStream)->Stop();
    }

TBool CAudioStreamEngine::Stop()
    {
    iOutputStream->Stop();  // will result in MaoscPlayComplete() call
    return EFalse;
    }
```

Note: On S60 3rd Edition, `CMdaAudioOutputStream::Stop()` can also be called from within `MaoscBufferCopied()`.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.