Archived:NSSTTS Common Symbian API

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*}}.

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

The TTS Common interface API defines common data types for TTS. The types cover text data segmentation and speech styles. The TTS Common module should be used with TTS Client Utility.

Header files

```
#include <nssttscommon.h>
```

Link against

```
LIBRARY nssttscommon.lib
```

Code examples

Simplest TTS code

```
...
// Header files (link against LIBRARY nssttsutility.lib and LIBRARY nssttscommon.lib)
#include <nssttsutility.h>
#include <nssttsutilityobserver.h>
...
class ... : public CBase, public MTtsClientUtilityObserver
{
...
protected:
    void MapcCustomCommandEvent( TInt aEvent, TInt aError );
    void MapcInitComplete(TInt aError, const TTimeIntervalMicroSeconds &aDuration);
    void MapcPlayComplete(TInt aError);
...
private:
...
    CTtsUtility* iTtsUtility;
...
}
...
    void ...:MapcCustomCommandEvent( TInt aEvent, TInt aError )
 {
 //
 }
    void ...:MapcInitComplete(TInt aError, const TTimeIntervalMicroSeconds &aDuration);
 {
```


(C) Copyright Nokia 2014. All rights reserved.

```
 //
 }
    void ...:MapcPlayComplete(TInt aError);
 {
 //
 }
...
#define SELECTED_PLUGIN_UID 0 // see
http://www.developer.nokia.com/Community/Wiki/Archived:NSSTTS_Utility_Symbian_API
       // for appropriate selection of pluginUids
 // Initialise the TTS utility
 iTtsUtility = CTtsUtility::NewL( *this );

 RArray<TUid> pluginUids;
 CleanupClosePushL( pluginUids );
 iTtsUtility->ListPluginsL( pluginUids );
 iTtsUtility->OpenPluginL( pluginUids.operator[]( SELECTED_PLUGIN_UID ) );
 CleanupStack::PopAndDestroy();

 TTtsStyle & iStyle=iTtsUtility->DefaultStyleL();
 iTtsUtility->SetDefaultStyleL(iStyle);

 iTtsUtility->OpenAndPlayDesL(_L("Hello! This is a sample of English Text. Now it's
12:05 PM"));
```

Notice that this sample allows selecting language and voice from the TTS phone menu (through DefaultStyleL/SetDefaultStyleL). Notice also that OpenPluginL is a necessary statement to load the NSTTS high quality voice set instead of the default low quality TTS. iTtsUtility->Close() is needed after each play, in order to release resources; besides, OpenPluginL shall be issued again after a Close(). Before issuing iTtsUtility->OpenDesL(_L("...")), add iTtsUtility->OpenAndPlayDesL(_L("")) to avoid system error 18.

```
TInt iVolume;
iVolume=0;
iTtsUtility->OpenPluginL( iOperator ); // Load plugin with voices
TTtsStyle & iStyle=iTtsUtility->DefaultStyleL(); // Get default TTS style
configuration...
iTtsUtility->SetDefaultStyleL(iStyle); // ...and load it
iTtsUtility->OpenAndPlayDesL(_L(""));  // Avoid system error 18
iTtsUtility->GetVolume(iVolume); // iVolume is valued with the default volume
configuration in TTS panel
iTtsUtility->OpenAndPlayDesL(_L("Hello world!"));
```

Notice that CTtsUtility* iTtsUtility and CMdaAudioPlayerUtility* m_pAudioPlayer can be both declared in the same class (e.g., m_pAudioPlayer = CMdaAudioPlayerUtility::NewL(*this, EMdaPriorityNormal, EMdaPriorityPreferenceNone)) to play sounds and clips with m_pAudioPlayer and TTS with iTtsUtility, sharing the same callbacks; volume can be controlled in MapcInitComplete through m_pAudioPlayer->SetVolume(...) and iTtsUtility->SetVolume(...).

Style information of TTS

The attributes that can be set include the synthesizing language, the age of the speaker, etc.

```
 TTtsStyle iStyle;
 TTtsStyleID iStyleId;
 // Use high quality TTS
 iStyle.iQuality = ETtsQualityHighOnly;
 // Use language and speaker set in Speech application.
```


(C) Copyright Nokia 2014. All rights reserved.

```
// Also speaking rate and volume is set according to Speech app.
iStyle.iLanguage = KTtsUndefinedLanguage;
// Add style
iStyleId = iTtsUtility->AddStyleL( iStyle );// where iTtsutility is CTtsUtility
```

Segment information of text

This class stores segment information.

```
TTtsSegment iSegment;
iSegment.SetStyleID( iStyleId ); // Sets the style for a segment.
//The style cannot be deleted during the lifetime of this object.
iSegment.SetTextPtr( iParsedText->Text() );//Sets a text pointer.
//The text cannot be deleted during the lifetime of this object.
```

CTtsParsedText

The CTtsParsedText class stores parsed text for TTS. The class owns the text and the corresponding phoneme sequence (for pattern recognition) and segmentation information.

Add Segment

Adds or inserts a new segment. The segments must be added in the correct order, that is, text and phoneme sequence pointers of segments follow each other. The segment can be deleted after the function returns.

```
CTtsParsedText* iParsedText = CTtsParsedText::NewL();
// Set text
iParsedText->SetTextL( KText1 );
iParsedText->AddSegmentL( iSegment, 0 );
```

Delete segment and number of segments

```
if ( iParsedText->NumberOfSegments() > 0 )
{
// Remove existing segment from parsed text structure
iParsedText->DeleteSegmentL( 0 );
}
```

Usage

```
{{ReviewForRemovalFromArchive|
'This article can be used to maintain legacy Symbian C++ applications exploiting TTS}}
```


(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.