ドキュメントトーカ日本語音声合成エンジンライブラリー

for Android

平成２４年1月１６日 第1．４版
平成２３年8月２５日 第1．３版
平成２３年2月１５日 第1．２版
平成２２年9月２４日 第1．１版
平成２２年7月２１日 第1．０版

クリエートシステム開発株式会社

2012/01/16　第 1.3 版　表音文字 JEITA 記述追加
2011/08/25　第 1.3 版　ライブラリー DTalkerTtsAndroidSDK.jar 追加
2011/02/15　第 1.2 版　ライブラリー DTalkerTtsEngine.jar 追加
2010/09/24　第 1.1 版　提供ファイルの変更
2010/07/21　第 1 版

当ソフトウエアの使用により生じた直接的、 または間接的損害（営業上の利益の損失、業務の中断、営業情報の喪失などによる損害を含む）については、一切責任を負いません。
ドキュメントトーカ日本語音声合成エンジンの波形処理部は富士通株式会社の音声合成技術を使用しております。言語処理部はクリエートシステム開発株式会社の技術を使用しております。

目次

１．概要 1

２．DTalkerTtsCntl　　jar 版 3

３．DTalkerTtsCntl クラス 3

３．１　Public Constructor 3
public DTalkerTtsCntl(Context context) 3
public DTalkerTtsCntl(Context context, Handler handler) 3

３．２　Public Method 4
openDTalkerTts 4
release 4

３．３　Sample 5

４．DTalkerTTSAndroidSDK 6

５．DTalkerTTS クラス 7

５．１　Constructor 7
public DTalkerTTS(Context context, Handler handler) 7
release 7

５．２　Speak controll 7
setEnable 7
speak 8
speakAt 8
speakPhoneme 8
sing 8
wavPlay 8
speakSyosai 8
speakSyosaiForIME 9
speakBookMark 9
makeWaveForText 9
makeWaveForSing 9
kanjiToKanaConvert 9
convertToSongPhoneme 9
stop 10
pause 10
resume 10
resumeNext 10
busy 10
isPause 10
setPositionTTS 10
getPositionTTS 11
setOffset 11
getOffset 11

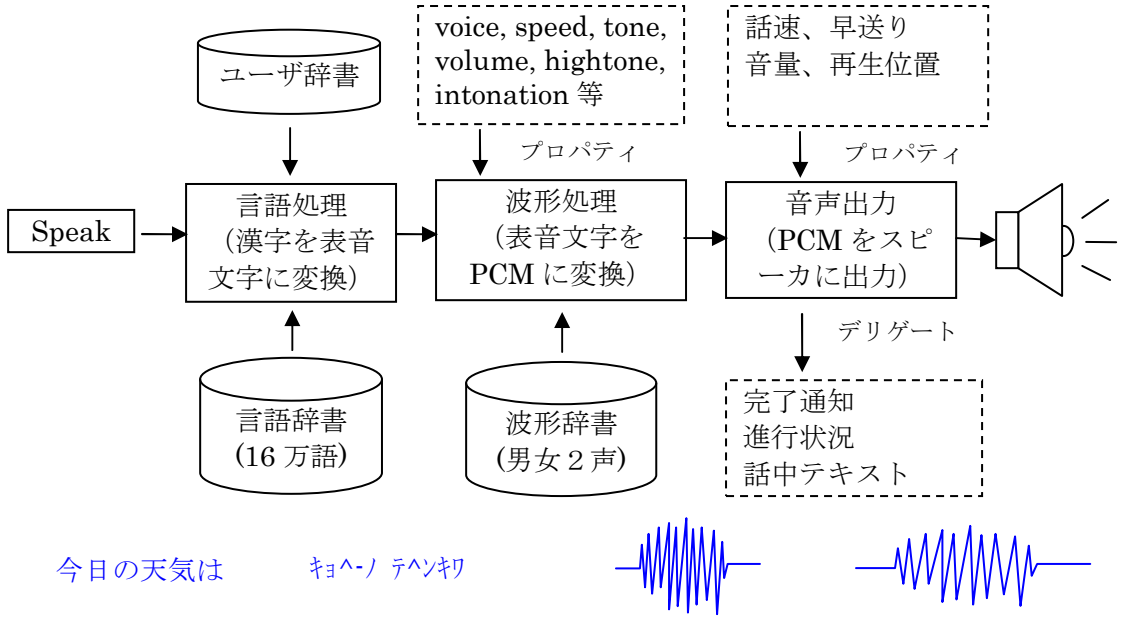
５．３　合成エンジンの設定 11
setVoice 11
getVoice 11

getVoiceName....11
setSpeed....12
getSpeed....12
setTone....12
getTone....12
setVolume....12
getVolume....12
setHightone....12
getHightone....13
setIntonation....13
getIntonation....13
setEcho....13
getEcho....13
setFastFoward....13
getFastFoward....13
setWarpRate....14
getWarpRate....14
setTempoRate....14
getTempoRate....14
setKigouYomi....14
getKigouYomi....14
setKutouten....15
getKutouten....15
setNumAnalysis....15
getNumAnalysis....15
setRomajiNumb....15
getRomajiNumb....15
setCrlfDelimtation....15
getCrlfDelimtation....16
setSpaceDelimtation....16
getSpaceDelimtation....16
setDefault....16
５．４　クリップボード読み上げ（サービス版のみ）....16
clipBoardSpeech....16
５．５　ユーザー辞書登録メソッド（サービス版のみ）....16
userDict(int speakable)....16
５．６　設定 Activity の呼び出しメソッド（サービス版のみ）....17
settings(int speakable)....17
６．イベントハンドラー....17
CALLBACK_DTS_FINISHED....17
CALLBACK_DTS_POSITION....17
CALLBACK_DTS_STRING....17
CALLBACK_DTS_OFFSET....18
CALLBACK_DTS_BOOKMARK....18
CALLBACK_DTS_DURATION....18
CALLBACK_DTS_STARTED....18
７．表音文字....19
８．歌唱用 MML 言語....21

１．概要

ドキュメントトーカ日本語音声合成エンジンは、正確な読み、自然なイントネーション、明瞭な発音を特徴とした日本語音声合成エンジンで、Windows や Mac 等で広く使用されております。
「ドキュメントトーカ日本語音声合成エンジン for Android」は、JNI(Java Native Interface）を使用し高速化を図るとともにシステムに負担をかけないようにバックグランドで動作するサービス（**DTalkerSpeechService**）として提供されます。
テキストを音声化するには、サービスに対して Speak を依頼することでバックグランドで発話が開始されます。

DTalkerSpeechService は以下の図のように Speak 依頼を受け付けると、言語処理、波形処理、そして音声出力というプロセスを踏みながら音声化されます。



言語処理：日本語の漢字混じり文を表音文字に変換する。
より正確な読みとアクセントを作り出すために以下の技術を使用しています。
・高精度な形態素解析と文法処理に基づく表音化技術
・約１６万語の単語の読みとアクセント等を記憶した単語辞書

波形処理：表音文字を音声波形（PCM）に変換する。
携帯端末での使用を考慮し、かつ、高い明瞭度（聞き取りやすさ）を実現するために以下の技術を使用しています。
・男女各約 15000 種の素片データに基づく１ピッチ波形編集方式
・声の高さ(5 段階)、速さ（10 段階)、音量(10 段階)

音声出力：音声波形を音声として出力する。
言語処理、波形処理を使用して、テキストをサウンドとして効率よく再生します。

サービス版のほか、単独で動作するライブラリー版（.jar）もご提供できますのでご相談ください。

開発環境
Android SDK
Java SE Development Kit
Eclipse IDE for java Developers
ADT Plugin for Eclipse

２．DTalkerTtsCntl　　jar 版

ドキュメントトーカ日本語音声合成エンジン for Android ライブラリー　jar 版は、
サービス版と違って、単独のライブラリーとしてリンクできるもので、他者からの利用ができません。

提供ファイル（ライブラリー版）

libs¥DTalkerTtsEngine.jar
libs¥armeabi¥libtts_dtalkerandroidtts.so
libs¥doc
javadoc
DTalkerDict.zip

Package 名

jp.co.createsysytem.DTalkerTtsCntl

使用方法

１．libs フォルダーを貴プロジェクトにコピーします。
２．Eclipse で参照ライブラリとして DTalkerTtsEngine.jar を設定します。
　Libs/DTalkerTtsEngine.jar を右クリック=>ビルドパス=>ビルドパスに追加
３．DTalkerDict.Zip を assets に追加します。
３．Eclipse に従いプログラミングします。
４．API は一部起動時の扱いを除きサービス版と同じです。

３．DTalkerTtsCntl クラス

サービス版との違いのみを記述しています。

３．１　Public Constructor

public DTalkerTtsCntl(Context context)

【パラメータ】　context コンテクスト

public DTalkerTtsCntl(Context context, Handler handler)

【パラメータ】　context　コンテクスト
　　handler　コールバックを受ける Handler
　　コールバックは Sample もしくはコールバックインターフェースを参照

３．２　Public Method

openDTalkerTts

【機能】	DTalkerTTs 辞書のオープン
【書式】	int openDTalkerTts()
【パラメータ】	なし
【戻り値】	1 ：正常 0 ：失敗
【説明】	辞書は/SDCARD/DTalkerDict 下にあるものとします。 なければ assets より DTalkerDict.zip を SDCARD に copy します。 assets になければ、失敗で戻ります。

release

【機能】	DTalkerTTsCntl をクローズします。
【書式】	void release()
【パラメータ】	なし
【戻り値】	なし
【説明】	辞書のクローズ、コントロールの後処理を行います。

以後は、後述 DTalkerTts とおなじメソッドが使用できます。
Jar 版には Activity の呼び出し、クリップボード読み上げ機能はございません。

３．３　Sample

```
{
   //イベントをハンドラーで受ける場合
   DTalkerTtsCntl tts = new DTalkerTtsCntl(this, mTtsHandler);
   int error = tts.openDTalkerTts();
   tts.speak( “12345” );
    ---
}
//ハンドラー
private Handler mTtsHandler = new Handler(){
   public void dispatchMessage(Message msg){
      if (msg.what==DTalkerTtsCntl.CALLBACK_DTS_FINISHED){
         Log.d("Sample","finished");
      }
      else if (msg.what== DTalkerTtsCntl.CALLBACK_DTS_STRING){
         Log.d("Sample","string="+msg.obj);
      }
      else if (msg.what== DTalkerTtsCntl.CALLBACK_DTS_OFFSET){
         Log.d("Sample","offset="+msg.arg1+",length="+msg.arg2);
      }
      else if (msg.what== DTalkerTtsCntl.CALLBACK_DTS_POSITION){
         Log.d("Sample","position="+msg.arg1);
      }
      else if (msg.what== DTalkerTtsCntl.CALLBACK_BOOKMARK){
         Log.d("Sample","book mark="+msg.obj);
      }
      else if (msg.what== DTalkerTtsCntl.CALLBACK_DURATION){
         Log.d("Sample","duration="+msg.arg1);
      }
      else{
         super.dispatchMessage(msg);
      }
   }
};
```

４．DTalkerTTSAndroidSDK

DTalkerTTSAndroidSDK は、ドキュメントトーカのサービス機能を利用するSDKであり、ドキュメントトーカがインストールされていればその音声機能を利用することができます。**ドキュメントトーカの製品版とデモ版を自動検出し切り分けて使用することができるSDKとして公開しております。**ドキュメントトーカがインストールされていなければ Nop として動作します。**これにより、Speechエンジンのインストールの有無を気にしないでアプリケーションを構築・配布できます。**

提供ファイル（サービス版）

DTalkerTTSAndroidSDK.zip

・DTalkerTtsAndroidSDK.jar
・DTalkerTtsDemo.apk
・DTalkerTtsAndroidSDK_Sample プロジェクト
・javadoc フォルダー
・Document フォルダー

Package 名

jp.co.createsysytem.DTalkerSDK

使用方法

１．「ドキュメントトーカ for Android」のインストール
「ドキュメントトーカ for Android」製品版 または、「デモ版」(同梱の DTalkerTtsDemo.apk)をインストールしておきます。
デモ版は、ADB を使い、 adb install DTalkerTtsDemo.apk でインストールできます。
初回、起動時にネットを介して辞書のダウンロードが行われますので、ネットワーク環境が動作するようにしておいてください。SDCARD にダウンロード後は、ネットワーク環境は必要ありません。

２．DTalkerTTSAndroidSDK.zip を解凍します。

３．「DTalkerTtsAndroidSDK.jar」ファイルをパソコンの任意の場所に置く。

３．Eclipse を起動しプロジェクトを起こします。

４．Android プロジェクトを作成する。ターゲットは Android 2.2 以上を指定すること。

５．パッケージエクスプローラで、作成したプロジェクトを選択し、右クリック＞「ビルド・パス」＞「外部アーカイブの追加」を選択する。

６．追加する外部アーカイブとして、「DTalkerTtsAndroidSDK.jar」ファイルを選択する。

７．パッケージエクスプローラで、プロジェクトの「参照ライブラリー」に「DTalkerTtsAndroidSDK.jar」が追加されていることを確認する

８．Javadoc の設定

(1) 配布ソフトウェアの「doc」フォルダをパソコンの任意の場所に置く。

(2) パッケージエクスプローラで、プロジェクトの「参照ライブラリー」＞「DTalkerTtsAndroidSDK. jar」を選択し、右クリック＞「プロパティ」を選択する。

(3) 表示された画面の左メニューから「Javadoc ロケーション」を選択する。

(4) 「Javadoc URL」が選択されていることを確認し、「Javadoc ロケーション・パス」に「doc」フォルダのパスを入力する。

５．DTalkerTTS クラス

DTalkerTTS クラスは、有償版、無償版を自動で切り替えてくれるクラスです。
サービスの Bind 処理なども組み込まれていますので、より簡単に使用することができ、合成エンジンの有無も気にすることがありません。
DTalkerTTSAndroisSDK(http://www.createsystem.co.jp/dtalkerAndroidSDK.html)として提供されています。

５．１　Constructor

public DTalkerTTS(Context context, Handler handler)

【パラメータ】	context　コンテクスト handler　コールバックを受ける Handler コールバックは Sample もしくはコールバックインターフェースを参照
【説明】	まず、製品版がインストールされていれば、製品版の Bind を行います。 インストールされていなければ、無償版の Bind を行います。 無償版もインストールされていなければ、handler に CALLBACK_DTS_STARTED, 3 が返ります。 Bind に成功すれば handler に、CALLBACK_DTS_STARTED, status がかえります。 Status:　3 = インストールされていない 2 = ネットから辞書をローディング中 1 = OK -1 = ネットワーク接続に失敗 -2 = Zip 解凍に失敗

release

【機能】	DTalkerTTS コントロールを開放します。
【書式】	void release()
【説明】	service connection を unbind します。

５．２　Speak controll

speak(),speakPhoneme(),sing(),wavPlay()はコールすると再生の開始・終了を待たずにリターンしてきますが、再生は呼ばれた順に再生されます。また、Stop()は、呼ばれたものすべてが中止されます。

setEnable

【機能】	speak 機能を有効にします
【書式】	void setEnable(boolean　enable)
【パラメータ】	enable　　true = speak 機能を有効とします。
【説明】	以下の関数に有効となります。 speak(), speakAT(), speakPhoneme(), sing(), wavPlay(), speakSyosai(), speakSyosaiForIME(), speakBookMark(),

speak

【機能】　指定した文章を喋る
【書式】　int speak(String textStr)
【パラメータ】　textStr　文章
【戻り値】　>0：正常
0 ：Error

speakAt

【機能】　指定した文章を喋る
【書式】　int speakAt(String textStr, int offset)
【パラメータ】　textStr　文章
offset　喋りだす位置
【戻り値】　>0：正常
0 ：Error

speakPhoneme

【機能】　指定した表音文字を喋る
【書式】　int speakPhoneme(String phonemeStr)
【パラメータ】　phonemeStr　表音文字
【戻り値】　>0：正常
0 ：Error

sing

【機能】　歌唱用 MML で記述されたテキストを音声で再生します。
【書式】　int sing(String mmlStr)
【パラメータ】　mmlStr　MML 言語
【戻り値】　>0：正常
0 ：Error
【説明】　MML 言語フォーマット：別紙参照

wavPlay

【機能】　wave file を再生する。
【書式】　int wavPlay(String fileName)
【パラメータ】　fileName　wave file 名
【戻り値】　>0：正常
0 ：Error
【説明】　wavefile のパーミッションによっては再生できません。

speakSyosai

【機能】　指定した文章を詳細読みで喋る
【書式】　int speakSyosai(String textStr)
【パラメータ】　textStr　文章
【戻り値】　>0：正常
0 ：Error

speakSyosaiForIME

【機能】	指定した文章を詳細読みで喋る
【書式】	int speakSyosai(String textStr)
【パラメータ】	textStr 文章
【戻り値】	>0：正常
	0 ：Error

speakBookMark

【機能】	指定した文章を BookMark として受け取ります。
【書式】	int speakBookMark(String textStr)
【パラメータ】	textStr 文章
【戻り値】	>0：正常
	0 ：Error
【説明】	再生される直前に Listener に callback が通知されます。実際の音声としての再生はされません。

makeWaveForText

【機能】	指定した文章を Wave ファイルに変換します。
【書式】	int makeWaveForText(String textStr, String fileName)
【パラメータ】	textStr 文章
	filename ファイル名
【戻り値】	>0：正常
	0 ：Error

makeWaveForSing

【機能】	指定した MML を Wave ファイルに変換します。
【書式】	int makeWaveForSing(String textStr, String filename)
【パラメータ】	textStr MML
	filename ファイル名
【戻り値】	>0：正常
	0 ：Error

kanjiToKanaConvert

【機能】	指定した文章を表音文字に変換します。
【書式】	String kanjiToKanaConvert(String textStr)
【パラメータ】	textStr 文章
【戻り値】	String 表音文字（歌唱用）

convertToSongPhoneme

【機能】	指定した文章を表音文字（歌唱用）に変換します。
【書式】	String kanjiToSongPhoneme(String textStr)
【パラメータ】	textStr 文章
【戻り値】	String 表音文字（歌唱用）
【説明】	表音文字から余計な記号を取り去ったものとなります

stop

【機能】	読み上げの中止
【書式】	void stop()
【パラメータ】	なし
【戻り値】	なし

pause

【機能】	読み上げの停止
【書式】	void pause()
【パラメータ】	なし
【戻り値】	なし

resume

【機能】	読み上げの再開
【書式】	void resume()
【パラメータ】	なし
【戻り値】	なし

resumeNext

【機能】	次の文節からの読み上げの再開
【書式】	void resumeNext()
【パラメータ】	なし
【戻り値】	なし

busy

【機能】	話中チェック
【書式】	boolean busy()
【パラメータ】	なし
【戻り値】	true: 話中

isPause

【機能】	読み上げ停止中の取得
【書式】	Boolean isPause()
【パラメータ】	なし
【戻り値】	true:停止中

setPositionTTS

【機能】	読み上げ開始位置の設定
【書式】	void setPositionTTS(int pos)
【パラメータ】	pos: 開始位置　speak()で指定されたtextStrの長さを100%としたとき開始位置を%で指定する。0-100
【戻り値】	なし
【説明】	現在発声中の呼び出しにかぎり有効。

getPositionTTS

【機能】	読み上げ位置の取得
【書式】	int getPosition()
【パラメータ】	なし
【戻り値】	speak()で指定された textStr の長さを 100%としたとき、読み上げ位置を%で返す。0-100
【説明】	現在発声中の呼び出しにかぎり有効。

setOffset

【機能】	読み上げ開始位置の設定
【書式】	void setOffset(int pos)
【パラメータ】	pos: 開始位置　speak()で指定された textStr の開始位置
【戻り値】	なし
【説明】	現在発声中の呼び出しにかぎり有効。

getOffset

【機能】	読み上げ位置の取得
【書式】	int getOffset()
【パラメータ】	なし
【戻り値】	speak()で指定された textStr の読み上げ位置を返す。
【説明】	現在発声中の呼び出しにかぎり有効。

５．３　合成エンジンの設定

setVoice

【機能】	声種の設定
【書式】	void setVoice(int voice)
【パラメータ】	voice 0:太郎, 1=花子, 2=小太郎, 3=花ちゃん 4=英語（米国）, 5=英語（英国）, 6=フランス語, 7=ドイツ語, 8=イタリア語,9=スペイン語
【戻り値】	なし
【説明】	0-3 は日本語。 外国語を指定した場合、日本語の読み上げは、前回指定されていた日本語で読み上げます。英文字が出現したときの状況により外国語に切り替わります。

getVoice

【機能】	声種の取得
【書式】	int getVoice()
【パラメータ】	なし
【戻り値】	int　0:太郎, 1=花子, 2=小太郎, 3=花ちゃん 4=英語（米国）, 5=英語（英国）, 6=フランス語, 7=ドイツ語, 8=イタリア語,9=スペイン語

getVoiceName

【機能】	声種の名称の取得

【書式】　　String getVoiceName()
【パラメータ】　なし
【戻り値】　　String 現在の声種の名称

setSpeed

【機能】　　声の速さを指定します。
【書式】　　void setSpeed(int v)
【パラメータ】　声の速さを 0-10 で指定。(6)
【戻り値】　　なし

getSpeed

【機能】　　声の速さを取得します。
【書式】　　int getSpeed()
【パラメータ】　なし
【戻り値】　　int 0-10

setTone

【機能】　　声の高さを指定します。
【書式】　　void setTone(int v)
【パラメータ】　声の高さを 1-5 で指定。(3)
【戻り値】　　なし

getTone

【機能】　　声の高さを取得します。
【書式】　　int getTone()
【パラメータ】　なし
【戻り値】　　int 1-5

setVolume

【機能】　　声の音量を指定します。
【書式】　　void setVolume(int v)
【パラメータ】　声の音量を 0-9 で指定。(9)
【戻り値】　　なし

getVolume

【機能】　　声の音量を取得します。
【書式】　　int getVolume()
【パラメータ】　なし
【戻り値】　　int 0-9

setHightone

【機能】　　声の High Tone を指定します。
【書式】　　void setHightone(int v)
【パラメータ】　0=normal tone、1=high tone　　(0)
【戻り値】　　なし

getHightone

【機能】	声の High Tone を取得します。
【書式】	int getHightone()
【パラメータ】	なし
【戻り値】	int 0-1

setIntonation

【機能】	声の抑揚を指定します。
【書式】	void setIntonation (int v)
【パラメータ】	声の抑揚を 0-3 で指定。 (1)
【戻り値】	なし

getIntonation

【機能】	声の High Tone を取得します。
【書式】	int getIntonation ()
【パラメータ】	なし
【戻り値】	int 0-3

再生中の音声パラメータの取得・設定をおこないます。
源音声の生成時と違い再生時に処理されるものですが、現在は生成時とほぼ同じタイミングとなります。

setEcho

【機能】	エコーOn/Off
【書式】	void setEcho (int v)
【パラメータ】	0：エコーオフ (0)
	1：エコーオン
【戻り値】	なし

getEcho

【機能】	エコーOn/Off 状態を取得します。
【書式】	int getEcho ()
【パラメータ】	なし
【戻り値】	int 0：エコーオフ, 1：エコーオン

setFastFoward

【機能】	早送り
【書式】	void setFastFoward (int v)
【パラメータ】	0=Normal, 1=倍速 (0)
【戻り値】	なし
【説明】	速度が 2 倍になりますがトーンも高くなります。

getFastFoward

【機能】	エコーOn/Off 状態を取得します。
【書式】	int getFastFoward ()
【パラメータ】	なし

【戻り値】　int 0：Normal, 1：倍速

setWarpRate

【機能】　声のトーンの比率を指定します。
【書式】　void setWarpRate (float v)
【パラメータ】　0.6-1.0　(1.0)
【戻り値】　なし
【説明】　0.6-1.0 の範囲でトーンを変更できます。トーンは現状(1.0)から高くなる方向で設定され、高くなると話速も速くなります。

getWarpRate

【機能】　声のトーンの比率を取得します。
【書式】　float getWarpRate ()
【パラメータ】　なし
【戻り値】　float 0.6 - 1.0

setTempoRate

【機能】　話速変換率を指定します。
【書式】　void setTempoRate(float tempoRate)
【パラメータ】　0.5 - 1.5　(1.0)
【説明】　現状(1.0)の話速から、0.5 から 1.5 の範囲で話速を変更できます。数字が小さいほど速くなり大きいほど遅くなります。なお、トーンは変更されません。　(1.0)
使用状況によっては、音質が悪くなったり、プチプチ音が出ることがあります。

getTempoRate

【機能】　話速変換率を取得します。
【書式】　float getTempoRate()
【パラメータ】　なし
【戻り値】　float 0.5 - 1.5

setKigouYomi

【機能】　記号をすべて読み上げるか、読み上げないかの設定を行います。
【書式】　void setKigouYomi(boolean v)
【パラメータ】　false= 読み上げない、true= 読み上げる　(false)
【戻り値】　なし

getKigouYomi

【機能】　記号をすべて読み上げるか、読み上げないかの取得を行います。
【書式】　boolean getKigouYomi()
【戻り値】　false= 読み上げない、true= 読み上げる
【パラメータ】　なし

setKutouten

【機能】	句読点を読み上げるか、読み上げないかの設定を行います。
【書式】	void setKutouten (boolean v)
【戻り値】	なし
【パラメータ】	false= 読み上げない、true= 読み上げる (false)

getKutouten

【機能】	句読点を読み上げるか、読み上げないかの取得を行います。
【書式】	boolean getKutouten ()
【パラメータ】	なし
【戻り値】	false= 読み上げない、true= 読み上げる

setNumAnalysis

【機能】	数字読み上げ解析を行うか行わないかの設定を行います。
【書式】	void setNumAnalysis (boolean v)
【戻り値】	なし
【パラメータ】	false=行わない、true= 行う (true)

getNumAnalysis

【機能】	数字読み上げ解析を行うか行わないかの取得を行います。
【書式】	boolean getNumAnalysis ()
【パラメータ】	なし
【戻り値】	false=行わない、true= 行う

setRomajiNumb

【機能】	ローマ字読みを行う文字数の設定を行います。
【書式】	void setRomajiNumb (int v)
【戻り値】	なし
【パラメータ】	0- (4)
【説明】	スペル読みするか、ローマ字読みするかの境界文字数を指定します。 例）aaa エーエーエー と読むか アアア と読むかの違い。

getRomajiNumb

【機能】	ローマ字読みを行う文字数の取得を行います。
【書式】	int getRomajiNumb ()
【パラメータ】	なし
【戻り値】	int 文字数

setCrlfDelimtation

【機能】	CRLF を文の境界とするかどうかの設定を行います。
【書式】	void setCrlfDelimtation (boolean v)
【戻り値】	なし
【パラメータ】	false=しない, true=する (true)

getCrlfDelimtation

【機能】　CRLF を文の境界とするかどうかの取得を行います。
【書式】　Boolean getCrlfDelimtation ()
【パラメータ】　なし
【戻り値】　boolean false=しない, true=する

setSpaceDelimtation

【機能】　SPACE を文の境界とするかどうかの設定を行います。
【書式】　void setSpaceDelimtation (boolean v)
【戻り値】　なし
【パラメータ】　false=しない, true=する　(true)
【説明】

getSpaceDelimtation

【機能】　SPACE を文の境界とするかどうかの取得を行います。
【書式】　Boolean getSpaceDelimtation ()
【パラメータ】　なし
【戻り値】　boolean false=しない, true=する

setDefault

【機能】　再生パラメータをデフォルトに設定します。
【書式】　void setDefault()
【パラメータ】　なし
【戻り値】　なし
【説明】　以下のプロパティが設定されます。
voice=0, speed=6, tone=3, volume=7, highTone=0, intonation=1
Echo=0, FF=0, warpRatio=1.0, tempoRatio=1.0,
Crlf=true, SPACE=false, Kigou=false, Kutouten=false, Num=true
Romaji=4

５．４　クリップボード読み上げ（サービス版のみ）

clipBoardSpeech

【機能】　クリップボード読み上げを行います。
【書式】　void　clipBoardSpeech(in boolean v)
【パラメータ】　true:読み上げ開始, false=読み上げ中止
【戻り値】　なし
【説明】　true を設定することでクリップボードにコピーされているテキストを読み上げます。一定時刻ごとにクリップボードの内容変化を検出し、変化があればそのテキストを読み上げます。

５．５　ユーザー辞書登録メソッド（サービス版のみ）

userDict(int speakable)

【機能】　ユーザー辞書登録 Activity を呼び出します。

【書式】　　void　userDict(int speakable)
【パラメータ】　0=Activity は発声不可, 1=Activity は発声可能
【戻り値】　　なし
【説明】　　ユーザー辞書登録 Activity を呼び出します。

５．６　設定 Activity の呼び出しメソッド（サービス版のみ）

settings(int speakable)

【機能】　　設定 Activity を呼び出します。
【書式】　　void　**settings** (int speakable)
【パラメータ】　0=Activity は発声不可, 1=Activity は発声可能
【戻り値】　　なし
【説明】　　設定 Activity を呼び出します。

６．イベントハンドラー

ハンドラーに通知されるメッセージの一覧

```
public static final int CALLBACK_DTS_FINISHED      = 0;
public static final int CALLBACK_DTS_STRING        = 1;
public static final int CALLBACK_DTS_OFFSET        = 2;
public static final int CALLBACK_DTS_POSITION      = 3;
public static final int CALLBACK_DTS_BOOKMARK      = 4;
public static final int CALLBACK_DTS_DURATION      = 5;
public static final int CALLBACK_DTS_STARTED       = 6;
```

CALLBACK_DTS_FINISHED

【機能】　　再生完了通知
【パラメータ】　なし
【説明】　　再生が完了したときに発生します。

CALLBACK_DTS_POSITION

【機能】　　再生位置の通知
【パラメータ】　arg1 = pos　　再生位置 0-100
【説明】　　再生位置の割合(0 - 100% full)が通知されます。

CALLBACK_DTS_STRING

【機能】　　話中テキストの通知
【パラメータ】　obj = (String)speakText　　再生中のテキスト
【戻り値】　　なし
【説明】　　再生中のテキストが通知されます。

CALLBACK_DTS_OFFSET

【機能】　話中テキストの通知
【パラメータ】　arg1 = offset
arg2 = length
【説明】　speak()メソッドで指定された textSrc の現在再生中の offset と length が通知されます。

CALLBACK_DTS_BOOKMARK

【機能】　BookMark の通知
【パラメータ】　obj = (String)textSrc
【説明】　speakBookMark()メソッドで指定された textSrc が再生される直前に通知されます。textSrc は発声はされません。

CALLBACK_DTS_DURATION

【機能】　再生時間の通知
【パラメータ】　arg1 = duration　再生時間 (ms)
【説明】　再生される直前に通知されます。

CALLBACK_DTS_STARTED

【機能】　DTalkerTts のローディング完了通知
【パラメータ】　arg1 = status
3 = インストールされていない
2 = ネットから辞書をローディング中
1 = OK
-1 = ネットワーク接続に失敗
-2 = Zip 解凍に失敗
【説明】　DTalkerTts のローディングが完了したときに発生します。
これ以後に、Speak 等が可能となります。

7．表音文字

半角のカタカナ及び制御文字列によって記述された文字列が表音文字列です。
一般的なテキストエディタで作成が可能であり、また、ライブラリを使用して自動的に作り出すことが可能です。
ドキュメントトーカでは、JEITA TT-6004で規定された表音文字のほか、独自の制御記号を使用しております。

表音文字列の一般的な作成手順
１．発音内容をカタカナで表記する。
２．アクセントをつける。
３．イントネーションをつける。

カタカナ表記の原則
普通にカタカナで表記してください。ただし、書き方と発音が違う場合があります。発音に忠実に書くのが最良の方法ですが、次のようなことに注意して書いてみてください。
アクセント：　アクセント位置には、〝^〟記号を入れる。
は と へ：　助詞の「は」と「へ」は「ワ」「エ」にかえる。
長音：　おとうさんは　ｵﾄ^ｰｻﾝﾜ
　　経験　ｹｰｹﾝ

ドキュメントトーカでは、アクセント記号は“^”となりますが、JEITA表音文字では、アクセントは以下の表のようになります。

“	ダブルクォート	非常に弱いアクセント
*	アスタリスク	弱いアクセント
‘	シングルクォート	通常のアクセント

ドキュメントトーカでは、弱アクセントなどは自動で変更されます。

鼻濁音※
「が・ぎ・ぐ・げ・ご」の発音を鼻にかかったような発音にする鼻濁音表記は ﾞ を ﾟ にかえる。ただし、JEITA表音文字では“&”を付加する。
例)「学校が」の場合　ｶﾞｯｺｰｶﾟ　または ｶﾞｯｺｰｶﾞ&

無声化文字※
無声化する文字には直後に # をつける。
出版　ｼｭ#ｯﾊﾟﾝ
学者　ｶﾞｸ#ｼｬ
JEITA表音文字では “%”を使用。

アクセントのつけかた
日本語のアクセントは文節毎に現れるので、まず文を文節毎に分解します。文節の分解のしかたとして簡単な方法としては、「ね」をつけてみて意味が通じるところを文節境界とする方法があります。また、句読点のあるところは必ずアクセント境界となります。
例）日本語の｜アクセントは｜文節毎に｜現れる｜音の｜高低で｜表される。｜

次に文節毎にアクセント位置を決定します。日本語の発声では、各文節の声の高さは、高と低の２種類しかなく、高から低へ変化する位置がアクセント位置となります。（声の高さがほぼ一定でアクセントのない平板型もあります。）

日本語の	ﾆﾎﾝｺﾞﾉ
アクセントは	ｱ^ｸｾﾝﾄﾜ
文節毎に	ﾌﾞﾝｾﾂｺﾞ^ﾄﾆ
現れる	ｱﾗﾜﾚ^ﾙ
音の	ｵﾄﾉ
高低で	ｺ^ｰﾃｰﾃﾞ
表される。	ｱﾗﾜｻﾚ^ﾙ。

表音文字列では、文節の区切りをスペースで表記し、アクセント位置を ^ で表します。
ﾆﾎﾝｺﾞﾉ ｱ^ｸｾﾝﾄﾜ ﾌﾞﾝｾﾂｺﾞ^ﾄﾆ ｱﾗﾜﾚ^ﾙ ｵﾄﾉ ｺ^ｰﾃｰﾃﾞ ｱﾗﾜｻﾚ^ﾙ。

イントネーションの付け方

イントネーションもアクセントと同じで声の高低で表されます。アクセントは文節単位という比較的短い区間での声の上げ下げであったのに対して、イントネーションは、より広い区間での声の上げ下げのことです。つまり、いくつかの文節をまとめて、発声のまとまりをつけることで、それには普通の文章を書くのと同様に句読点を入れることで指定が出来ます。句読点の間隔が長くなると発音が低くなってしまいます。長すぎる区間の中間にスラッシュ（／）を入れることにより、声を立て直すことができます。その目安としてカタカナで１５～２０文字程度が適当です。合成ドライバでは自動的に立て直すアルゴリズムをとっていますので、この件に関してはあまり気にすることはありません。
疑問調の文には最後に疑問符を付ければ、尻上がりの発声にすることができます。

８．歌唱用 MML 言語

歌詞データの一般的なフォーマットを下記に示します。

歌詞データ１,歌詞データ２,歌詞データ３,――――,歌詞データｎ改行

歌詞データｎは以下のフォーマットになります。

ＭＭＬ音符データ　{ｽﾍﾟｰｽ 歌詞}

頭が#で始まる行はコメント行でその行は無視されます。

ＭＭＬ音符データ

音程

Ann, Bnn, Cnn, Dnn, Enn, Fnn, Gnn

ラ　シ　ド　レ　ミ　ファ ソ

A～G までではそれぞれ音階をあらわし、nnは音長を意味します。

音長は省略可能で、省略した場合はＬで指定された長さになります。

nn=１は全音符、２は２分音符、４は4分音符、８は８分音符、１６は１６分音符を意味します。

音程の後ろにはスペースを挟んで歌詞を指定します。

例）C4 ど,D4 れ, E8 み

休符

Rnn

nnは音長を意味します。音程の場合と同様です。

テンポ

Tmmm

mmmは1分あたりの4分音符の個数をあらわします。

音長

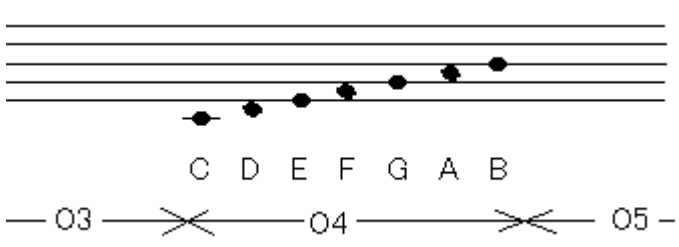
Lnn

デフォルトの音長を定義します。nnが4の場合には4分音符をデフォルトとし、音程表記のときの音長を省略できます。

オクターブ

03, 04, 05

04が標準のオクターブとなります。そのときの C,D,E,F,G,A,B が 04のﾄﾞﾚﾐﾌｧｿﾗｼﾄﾞ となります。

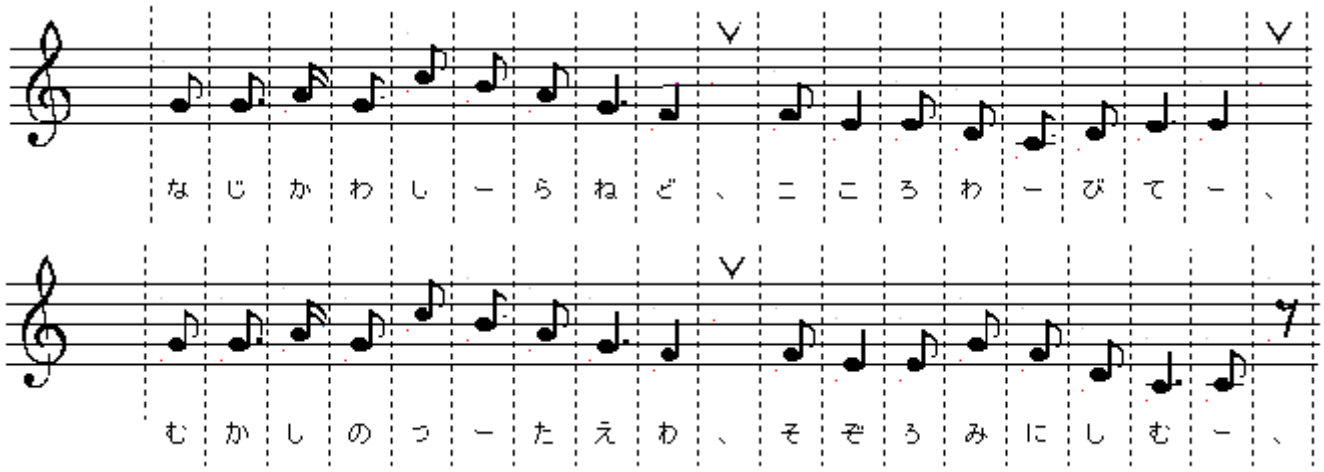


例）04,C4 ど,03,B4 し,A4 ら, 04,C4 ど

オクターブアップ
　＜　例）O3,B4 し,＜,C4 ど
オクターブダウン
　＞
タイ（同一音程でのみ有功）
　＆　例）C4 ど,＆,C4 ど
ブレス（息継ぎ）
　＾　例）C4 ど,＾,C4 ど
半音上げ
　＃　例）C8＃　ど
　＋
半音下げ
　－　例）ｃ8－ ど

歌詞
　歌詞は全角ひらがな、カタカナで表記します。音符1個につき１文字から２文字を割り当てます。

例）



T60
O4,G8 な,G8. じ,A16 か,G8 わ,O5,C8 し,O4,B8 ー,A8 ら,G4. ね,F4 ど,＾,F8 こ,
E4 こ,E8 ろ,D8 わ,C8 ー,D8 び,E4. て,E4 ー,＾,
O4,G8 む,G8. か,A16 し,G8 の,O5,C8 つ,O4,B8 ー,A8 た,G4. え,F4 わ,＾,F8 そ,
E4 ぞ,E8 ろ,G8 み,F8 に,D8 し,C4. む,C8 ー,R8 ,

平成２４年１月１６日　第1.4版
平成２３年８月２５日　第1.3版
平成２３年２月１５日　第1.2版
平成２２年９月２４日　第1.1版
平成２２年７月２１日　第1.0版

日本語音声合成ライブラリ for Android ライブラリー

クリエートシステム開発株式会社

〒190-0012　東京都立川市曙町２－４－４
昭和ビル７F
TEL：　０４２－５２７－５７７２
FAX：　０４２－５２７－５０７２
E-Mail：　sup-info@createsystem.co.jp
ＵＲＬ：http://www.createsystem.co.jp