

AquesTalk1 Android マニュアル

株式会社 アクエスト
www.a-quest.com

概要

本文書は、音声合成ライブラリ AquesTalk1 Android をアプリケーションに組み込んで使用するためのプログラミングの方法、注意点を示したものです。

AquesTalk1 は、かな表記の音声記号列から WAV 音声データを生成するライブラリです。

声種の変更は so ライブラリを差し替えて行います。

本ライブラリを使用するには、開発ライセンスキーの設定が必要です。このライセンスキーを設定しない場合は、評価版として動作し、以下の制限があります。

評価版の制限 「ナ行、マ行」を指定すると、すべて「ヌ」と発声します

また、本ライブラリをアプリケーションに組み込んで使用する際には**使用ライセンス**、配布には**頒布ライセンス**が必要です。ライセンスの種類や購入方法は、弊社サイトのライセンスのページを参照してください。

仕様

ライブラリ形式	So 形式共有ライブラリ (JNI 実装) 各種 ABI に対応 armeabi-v7a, arm64-v8a, x86, x86_64, riscv64
対応 OS	Android 5 (API 21)以降
入力データ形式	かな表記音声記号列 (UTF8)
出力データ形式	WAV フォーマット (8KHz サンプリング, 16bitPCM, モノラル)
声種	9 種(so ライブラリ差し替え)
関数 I/F	Java/JNI
開発環境	Android NDK r29
ライブラリサイズ	約 110KByte

関数 API

syntheWav

package AquesTalk

説明	かな表記音声記号列(SJIS)から音声波形を生成します
構文	byte[] syntheWav(String kanaText, int speed)
引数	
<i>kanaText</i>	音声記号列(UTF8)を指定
<i>speed</i>	発話速度[%] 50-300 の間で指定 デフォルト:100 値を大きく設定するほど、速くなる
戻り値	wav フォーマットのデータ エラー時には、長さ1で、先頭にエラーコードが返る。

setDevKey

package AquesTalk

説明	開発ライセンスキーを指定します。 音声波形を生成する前に一度呼び出すことで、以降、製品版とし動作し、評価版の制限がなくなる。
構文	int setDevKey(String key)
引数	
key	開発ライセンスキー
戻り値	ライセンスキーが正しければ 0、正しくなければ 1 が返る。 不正なキーでも 0 を返す場合がある。このとき制限は解除されない。

setUsrKey

package AquesTalk

説明	使用ライセンスキーを指定します。 音声波形を生成する前に一度呼び出すことで、以降、音声透かしが使用ライセンス無しから取得済みに変化する。
構文	int setUsrKey(String key)
引数	
key	使用ライセンスキー
戻り値	ライセンスキーが正しければ 0、正しくなければ 1 が返る。 不正なキーでも 0 を返す場合がある。このときはライセンス無しのままである。

声記号列

AquesTalk1 は、かな表記の音声記号列から音声を作成します。漢字を含んだテキスト文字列から音声を作成するときは、別途、言語処理ライブラリ AqKanji2Koe を用いて漢字仮名交じりテキストから音声記号列に変換する必要があります。

音声記号列の詳細は、付属の音声記号列仕様書を参照してください。

※AquesTalk1 Ver.2.0 から「フュ」などの音声記号列の音韻が拡張されています。

エラーコード表

関数が返すエラーコードの内容は、次の通りです。

値	内容
100	その他のエラー
101	メモリ不足
105	音声記号列に未定義の読み記号が指定された
105	音声記号列に未定義の読み記号が指定された
106	音声記号列のタグの指定が正しくない
107	タグの長さが制限を越えている(または[>]が見つからない)
108	タグ内の値の指定が正しくない
200	音声記号列が長すぎる
201	1つのフレーズ中の読み記号が多すぎる
202	音声記号列が長すぎる
203	ヒープメモリ不足
204	音声記号列が長すぎる

ビルド・実行

ファイル

ビルド及び実行時に下記のファイルを使用します。

libAquesTalk.so : AquesTalk1 ライブラリの実体（ネイティブコード）

AquesTalk.java : libAquesTalk.so を Java からアクセスするためのラッパークラス

任意のアプリケーションを開発するときは、libAquesTalk.so と AquesTalk.java ファイルを、アプリケーションのプロジェクトに追加します。Android Studio の環境であれば、プロジェクトディレクトリに、上のファイルを下記のようなディレクトリ構造上に(ディレクトリがなければ作成)コピーします。

「AquesTalk1 Android」の実体はネイティブコードで書いてあり、JNI 経由で呼び出しています。そのため、使用するシステム環境（ABI）に応じたライブラリを使用する必要があります。たとえば、使用端末が ARM CPU の場合は arm64-v8a か armeabi-v7a を、Intel CPU の場合は x86, x86_64 などのライブラリを使用します。

Android では FAT-Binary(シングル APK)によって、複数の異なる ABI のライブラリを 1 つの apk に含めることができ、動作マシン側で自動的に選んで実行します。そのため、apk のサイズさえ気にしなければ、SDK パッケージの jniLibs 以下すべてをアプリのプロジェクト側にコピーします。

```
<app/src/main>
  /java
    <アプリのソース>
    /questalk
      /AquesTalk.java
  /jniLibs
    /arm64-v8a/libAquesTalk.so
    /armeabi-v7a/libAquesTalk.so
    /x86/libAquesTalk.so
    /x86_64/libAquesTalk.so
  /res
```

なお、呼び出しの関係は、次のようになります。

* アプリ -> AquesTalk.java -> libAquesTalk.so

アプリのソースコードへの実装方法

任意のアプリに AquesTalk1 を組み込む方法を、本 SDK に含まれているサンプルアプリのコードを参照しながら以下に示します。（samples¥AqTkApp¥app¥src¥main¥java¥questalk¥AqTkApp.java）

AquesTalk のクラスをインポートします。

```
import questalk.AquesTalk;
```

AqTkApp.java では、アプリ起動時の onCreate() のなかで、インスタンスを生成しています。

```
questalk = new AquesTalk();
```

そして、発声ボタンが押されたときに、音声記号列 と発話速度を取得して、次のように音声データをメモリ上に生成します。なお、音声記号列の文字コードは UTF-8 で指定します（Android のデフォルト）。

```
onPlayBtn():  
    byte[] wav = aquestalk.syntheWav(koe, speed);
```

ここで得られる音声データは wav フォーマット（8KHz,Mono,Straight PCM）です。
音声記号列が不正な場合など生成エラーの時は、byte[] のサイズが 1 になり、配列の先頭にエラーコードが返されます。エラーコードは、エラーコード表を参照ください。

音声を出力するメソッドは用意されていませんので、音声再生はアプリ側で記述する必要があります。
AudioTrack クラスを用いて音声出力するコードの例がサンプルアプリ中の AqTkApp.java にありますので、参考にしてください。

注意すべき点は、AquesTalk1 の出力は wav フォーマットであるのに対し、AudioTrack で再生するデータは、ヘッダを含まない生のデータ列を指定する必要があります。そこで、生成した音声データの先頭にある WAV ヘッダ(44 バイト)を除いて、それ以降を AudioTrack に渡します。また、データの長さ（サンプル数）は、1つのサンプルが 2 バイトですので、次のようになります。

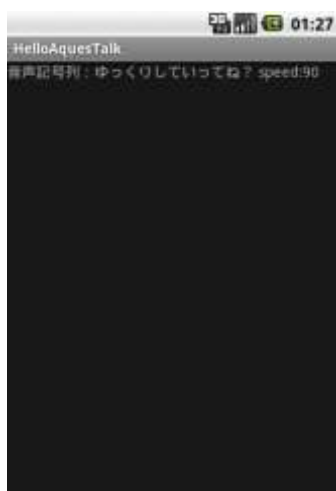
```
サンプル数 = (wav.length-44) / 2
```

サンプルプログラム

本パッケージには、ライブラリと一緒に、サンプルアプリが 2 つ入っています。 サンプルアプリのソースコードは自由に改変してご使用ください。 アプリのプロジェクトに AquesTalk1 ライブラリ(.so)は含まれていませんので、上記の「ファイル」を参考に指定のフォルダにコピーして使用してください。

* HelloAquesTalk: 最も単純なサンプルで、起動時に規定のメッセージを合成出力するだけのものです。

* AqTkApp: GUI を持たせ、任意の音声記号列と話速を指定して音声合成するものです。



アプリ開発ガイドライン

アプリケーションの開発(評価での使用を除く)は、以下のガイドラインに従ってください。

ライセンスキー

本ライブラリの動作は、開発ライセンスキーと使用ライセンスキー、頒布ライセンスキーの3種類の関連キーに依存します。これらのキーは、各ライセンス購入時に発行されるライセンス証に記載されています。

`setDevKey()` をアプリケーションの起動初期に一度呼び出します。引数には開発ライセンスキーを指定します。これにより製品版として動作し、評価版の制限がなくなります。

`setUsrKey()` をアプリケーションの起動初期に一度呼び出します。`setDevKey()` との呼び出し順序は任意です。引数には、使用ライセンスキー、または頒布ライセンスキーを指定します。
この指定により、合成音声データに含まれる透かしが、使用ライセンス無しの状態から取得済みに変化します。この変化による聴感上の違いはありません。

頒布ライセンスによりアプリを配布する場合は、頒布ライセンスキーを指定して呼び出します。
それ以外の場合は、エンドユーザが使用ライセンスキーを指定できるようにします。なお、エンドユーザが個人かつ非営利の利用の場合は使用ライセンスが不要なので、使用ライセンスキーが未指定の場合は、この関数の呼び出しをスキップして構いません。
関数の戻り値のチェックは必ず行い、エラーの場合はエンドユーザにその旨を通知してください。

文書履歴

日付	版	更新内容
2011/01/31	1.0	新規作成
2016/03/11	1.4	SDK Ver. 1.4、IDE を AndroidStudio に変更
2019/05/29	1.4.1	Android Studio 3.4 用に修正
2025/04/11	2.0	AquesTalk1 Ver. 2.0 用に関数 IF、サンプル等の記載を全面改訂 サンプルアプリ作り直し