# AquesTalk10 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 AquesTalk10 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。

AquesTalk10 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。

AquesTalk10 は、声質パラメータを指定することで様々な声質の音声を生成できる特徴があります。

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

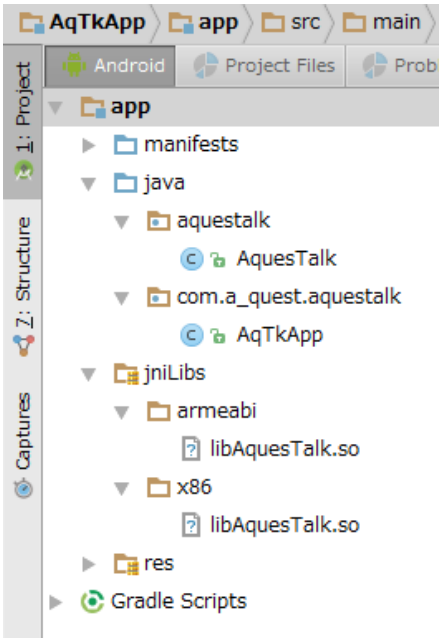
| | |
|---|---|
| ライブラリ形式 | so 形式 ダイナミックライブラリ（JNI 実装）<br>armeabi, armeabi-v7a, arm64-v8a, mips, mips64, x86, x86_64 |
| 対応 OS | Android 2.3 (API レベル9) 以降 |
| 入力データ形式 | かな表記音声記号列（UTF8) |
| 出力データ形式 | WAV フォーマット（16KHz サンプリング,16bitPCM,モノラル）データ<br>*サンプリング周波数は声質パラメータにより変化 |
| 声種 | ベース音素片 3 種　パラメータでユーザカスタマイズ可 |
| 関数 I/F | Java クラス呼び出し(ライブラリ内部でネイティブ呼び出し) |
| ライブラリサイズ | 約 350KByte (アーキテクチャあたり) |
| 開発環境 | Android NDK r10e |

## 使用方法

### アプリのプロジェクトに AquesTalk ライブラリを組み込む

下記の 2 ファイルをプロジェクトに設定(組み込み)します。

| | |
|---|---|
| libAquesTalk.so | AquesTalk 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 |




アプリケーションを開発するときは、libAquesTalk.so と (src/aquestalk/)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、上のファイルを図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。

libAquesTalk.so はネイティブコードのため、使用するマシン環境（ABI)に依存します。本 SDK の libs フォルダから、使用環境に応じたライブラリを選択してプロジェクトに追加します。たとえば、使用端末が ARM CPU の場合は arm64-v8a, armeabi, armeabi-v7a を、Intel CPU の場合は x86 などのライブラリを使用します。

なお、1 つの apk に複数の ABI のライブラリをに含めたアプリを作成できます。FAT-Binary(シングル APK)と呼ばれ、例えば、図のように jniLibs に armeabi と x86 の 2 つのライブラリを配置してビルドするだけで ARM と Intel CPU の両方に対応したアプリが作成できます。

なお、アプリからライブラリの呼び出し関係は、次のようになります。
　　アプリ　-> AquesTalk.java -> libAquesTalk.so

### プログラムコードへ追加

SDK に入っているサンプルアプリのコードをもとに、コードへの組み込み例を示します。
(samples¥AqTkApp¥app¥src¥main¥java¥com¥a_quest¥aAquestTalk¥AqTkApp.java)

23 行：AquesTalk　クラスをインポートします。
26 行：AquesTalk クラスの変数を宣言します。
38 行：new で AquesTalk オブジェクトを生成します。この例では onCreate()で生成していますが、音声合成の都度、生成することもできます。
41 行からは、「発声」ボタンが押されたときに呼び出されるメソッドです。
43 行：声種を設定します。ここでは、F1 を指定しています。なお、デフォルトの声種も F1 です。
44～46 行：声質を設定します。スライダーの値を取得し、クラス変数の声質パラメータにセットしています。各声質パラメータの値の範囲は AquesTalk.java に規定されています。スライダーの値に 50 や 20 を加算しているのは、スライダーコントロールの最小値が常に 0 という制限のためです。
50 行:テキストボックスから取り出した音声記号列を引数として onPlayBtn メソッドをコールしています。

62 行：ここで、AquesTalk のメソッドである synthe を呼び出して音声を合成します。引数は音声記号列の文字列のみです。戻り値は WAV フォーマットの音声データの入った byte 配列です。

63 行：synthe メソッドは、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、下章のエラーコード表を参照ください。

67 行～:AudioTrack クラスを用いて音声を出力します。出力に使用するクラスはいずれでも構いません。また、音声をファイル出力する場合は、この部分で wav を書き出すこともできます。
注意すべき点は、AquesTalk 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。
73 行：サンプリング周波数をセットします。WAV ヘッダの中の情報からサンプリング周波数を生成しています。
76 行:音声データのサイズ[byte]をセットします。WAV ヘッダのサイズ 44byte を差し引いたサイズです。
78 行：audioTrack に音声データを書き込みます。
79 行：音声出力を開始します。

AqTkApp.java より

```
...
23  import aquestalk.AquesTalk;       // AquesTalk クラス 別途 libAquesTalk.so が必要
24
25  public class AqTkApp extends Activity {
26      AquesTalk aquestalk;
27      AudioTrack audioTrack;
28
29      /**
30       * Called when the activity is first created.
31       */
32      @Override
33      public void onCreate(Bundle savedInstanceState) {
34          super.onCreate(savedInstanceState);
...
38          aquestalk = new AquesTalk();
39          Button playButton = (Button) findViewById(R.id.button_synthe);
40
41          playButton.setOnClickListener(new View.OnClickListener() {
42              public void onClick(View view) {
43                  aquestalk.SetPreset(AquesTalk.PRESET.F1);
44                  aquestalk.spd = ((SeekBar) findViewById(R.id.spd)).getProgress() + 50;
45                  aquestalk.pit = ((SeekBar) findViewById(R.id.pit)).getProgress() + 20;
46                  aquestalk.lmd = ((SeekBar) findViewById(R.id.lmd)).getProgress();
47                  EditText textKoe = (EditText) findViewById(R.id.text_koe);
48                  String koe = textKoe.getText().toString();
49                  onPlayBtn(koe);
50              }
51          });
52      }
...
60      private void onPlayBtn(String koe) {
61          // 音声合成
62          byte[] wav = aquestalk.synthe(koe);
63          if (wav.length == 1) {//生成エラー時には,長さ 1 で、先頭にエラーコードが返される
64              Log.v("AQTKAPP", "AquesTalk Synthe ERROR:" + wav[0]);
65              Toast.makeText(this, "音声記号列が正しい？ :" + wav[0], Toast.LENGTH_LONG).show();
66          } else {      // 音声出力
67              if (audioTrack != null) {// インスタンスがあれば停止/解放
68                  audioTrack.stop();
69                  audioTrack.release();
```

```
                }
                audioTrack = new AudioTrack(
                        AudioManager.STREAM_MUSIC,
                        (wav[25] << 8) + wav[24], // WAV ヘッダからサンプリング周波数[Hz]を取得
                        AudioFormat.CHANNEL_OUT_MONO,
                        AudioFormat.ENCODING_PCM_16BIT,
                        wav.length - 44,
                        AudioTrack.MODE_STATIC);
                audioTrack.write(wav, 44, wav.length - 44);// 44:wav のヘッダサイズ
                audioTrack.play();
            }
        }
...
```

## サンプルアプリの実行

SDK 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。

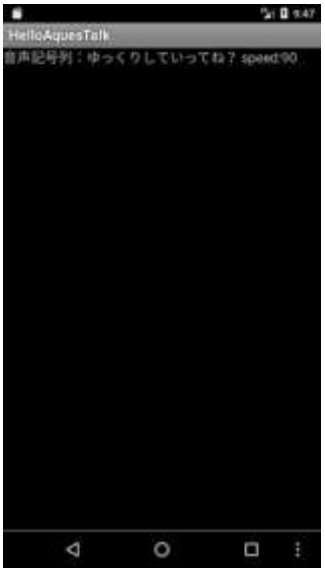
* HelloAquesTalk：
  最も単純なサンプルで、起動時に規定のメッセージを合成出力するだけのものです。

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

ビルド・実行には Android Studio の開発環境が構築されていることが前提です。

1. SDK samples フォルダの HelloAquesTalk または AqTkApp フォルダを任意の場所にコピー
2. Android Studio を起動
3. menu>File>Open でコピーした HelloAquesTalk または AqTkApp フォルダを Open。
4. 各種設定ファイルや build が自動的に行われます。
5. menu>Run>Run ’app’で実行します。

なお、開発ライセンスキーを設定していないので、評価版の制限で「ナ行、マ行」が、すべて「ヌ」になります。




# クラス API

## フィールド

**int bas**　声質パラメータ：基本素片　F1E/F2E/M1E (または 0/1/2)

**int spd**　声質パラメータ：話速　　50-300 default:100

**int vol**　声質パラメータ：音量　　0-300 default:100

**int pit**　声質パラメータ：高さ　　20-200 default:基本素片に依存

**int acc**　声質パラメータ：アクセント　0-200 default:基本素片に依存

**int lmd**　声質パラメータ：音程１　　0-200 default:100

**int fsc**声質パラメータ：音程２(サンプリング周波数) 50-200 default:100

## コンストラクタ

**public AquesTalk()**

AquesTalk オブジェクトを生成。声質はデフォルト(声種 F1)。

**public AquesTalk(int preset)**

AquesTalk オブジェクトを生成。引数に声種を指定。

**パラメータ：**
preset　　声種。PRESET.F1 などを指定。

## メソッド

**public byte[] synthe(String kanaText)**

音声を合成し、WAV 形式のデータを返す。

**パラメータ：**
kanaText　　音声記号列を指定。

**戻り値：**
WAV フォーマットの音声データ。エラーの場合は、配列のサイズが１でエラーコードが入る。

**public void SetPreset(int preset)**

指定の声種で声質パラメータをセット。

**パラメータ：**
preset　声種。PRESET.F1 などを指定。

## クラスメソッド

**public static int setDevKey(String devkey)**

開発ライセンスキーをセット。アプリの最初に一度だけ呼び出す。正常終了すると製品版とし動作し、評価版の制限がなくなる。

**パラメータ：**
devkey　開発ライセンスキー文字列(半角英数)

**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。

**public static int setUsrKey(String usrkey)**

使用ライセンスキーをセット。アプリの最初に一度だけ呼び出す。正常終了すると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から取得済みに変化する。

**パラメータ：**
usrkey　使用ライセンスキー文字列(半角英数)

**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。

## C 関数インターフェース

libAquesTalk.so には、Unity などからのアクセスのために、以下の C の関数のエントリポイントも用意してあります。
AquesTalk_Synthe、AquesTalk_Synthe_Utf8、AquesTalk_Synthe_Utf16、AquesTalk_FreeWave、AquesTalk_SetDevKey、AquesTalk_SetUsrKey

引数、戻り値、動作についての詳細は、AquesTalk10 Win マニュアルを参照してください。

## 音声記号列

AquesTalk10 は、かな表記の音声記号列から音声を合成します。漢字を含んだテキスト文字列から音声を合成するときは、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。

## 声質パラメータ

AquesTalk10 は声質パラメータの値を変更することで、様々な声種で合成できます。声質パラメータの種類と効果を以下に示します。

| | |
|---|---|
| **bas** | 声色のベースとなる基本素片 F1E,F2E,M1E のいずれかを指定 |
| **spd** | 話速。値が大きいほど発話速度が速い |
| **vol** | 音量。デフォルトは 100。値が大きいほど音量が大きくなる。100以下を指定の場合は、比例して音量が変化する。100以上を指定した場合は、コンプレッサーが機能する。 |
| **pit** | 声の高さがパラメータに比例する。 |
| **acc** | アクセントの強さ。値が大きいほどアクセントによるピッチの高低が大きくなる。 |
| **lmd** | 主に声質の高低を表現するが、より複雑な声質の変化がある。 |
| **fsc** | 声質の高低を表現。デフォルトは 100。サンプリング周波数を変化するだけなので、これに応じて話速や声の高さも同時に変化する。 |

AquesTalk.java には、声質パラメータの値をセットしたプリセット声種を複数用意しています。簡単に使う場合は、これを AquesTalk()や SetPreset()の引数に指定して設定することができます。

## エラーコード表

メソッド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 |
| 103 | 音声記号列指定エラー(語頭の長音、促音の連続など) |
| 104 | 音声記号列に有効な読みがない |
| 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[>]がみつからない) |
| 108 | タグ内の値の指定が正しくない |
| 120 | 音声記号列が長すぎる |
| 121 | 1つのフレーズ中の読み記号が多すぎる |
| 122 | 音声記号列が長い(内部バッファオーバー1) |

# 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。

AquesTalk.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。

AquesTalk.setUsrKey()をアプリケーションの起動初期に一度呼び出します。AquesTalk.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、このメソッドの呼び出しをスキップして構いません。
メソッド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。

## 声質パラメータ

エンドユーザによる声質パラメータの変更機能の有無は任意です。

変更可能にする場合、アプリケーション間での声質パラメータの値を共有するために、声質パラメータの値を写像しないでください。
例えば、話速のパラメータの範囲は 50 から 300 ですが、これを 0 から 100 の範囲にマッピングしてエンドユーザに提示しないでください。

エンドユーザに提示する各パラメータの名称には、AquesTalk.java に記載の漢字名称(話速など)や、3 文字の構造体変数名(spd など)を用いるのが望ましいです。

# 文書履歴

| 日付 | 版 | 変更箇所 | 更新内容 | 更新者 |
|---|---|---|---|---|
| 2017/10/22 | 1.0 | | 新規作成 | N.Y |
| 2017/10/27 | 1.01 | | C の関数エントリポイントの記述追加 | N.Y |