AquesTalk1 Mac 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 Mac 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 dylib 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	dylib 形式共有ライブラリ Arm64
動作環境	Apple シリコン搭載 Mac (Intel Mac は非対応)
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
関数 I/F	C 関数呼び出し
マルチスレッド	対応
ライブラリサイズ	約 150KByte
外部依存ライブラリ	/usr/lib/libc++.1.dylib、/usr/lib/libSystem.B.dylib

関数 API

AquesTalk_Synthe_Utf8　　AquesTalk1-mac.h

説明	かな表記音声記号列(UTF-8)から音声波形を生成します
構文	unsigned char * **AquesTalk_Synthe_Utf8**(const char **koe*, int *speed*, int **size*)
引数	
koe	音声記号列（UTF8 NULL 終端　BOM は付与しない）を指定
speed	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
size	生成した音声データのサイズが返る[byte]（エラーの場合はエラーコードが返る）
戻り値	WAV フォーマットの音声データを返す。 ヒープ領域を関数内部で確保するので、解放は AquesTalk_FreeWave()で行う。 エラー時は、NULL を返す。このとき size にエラーコードが設定される。

AquesTalk_FreeWave　　AquesTalk1-mac.h

説明	音声データの領域を開放
構文	void **AquesTalk_FreeWave**(unsigned char **wav*)
引数	
wav	WAV フォーマットのデータ（AquesTalk_Synthe()等で生成した音声データ）
戻り値	なし

AquesTalk_SetDevKey　　AquesTalk1-mac.h

説明	開発ライセンスキーを設定。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 **AquesTalk_SetDevKey**(const char **key*)
引数	
key	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。

説明	使用ライセンスキーを設定。音声波形を生成する前に一度呼び出すことで、以降、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。
構文	int **AquesTalk_SetUsrKey**(const char **key*)
引数	
key	使用ライセンスキー、または頒布ライセンスキーの文字列(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、まれに 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。

音声記号列

AquesTalk1 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。

※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[>]がみつからない)
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

サンプルプログラム

AqTkCmd と HelloAqTk1 の 2 つのサンプルプログラム(Xcode プロジェクト)が入っています。
動作確認環境は Xcode V.16.3、macOS 15.4.1 です。

AqTkCmd は C++で記述されたコマンドラインアプリです。音声記号列を入力し WAV データを出力します。

```
$ echo こんにちは | ./AqTkCmd > output.wav
```

HelloAquesTalk は Swift で記述されたアプリです。エディットボックスに音声記号を入力し、ボタンの押下でスピーカから合成音声を出力します。

プログラミングガイド

ポイントは、ヘッダファイルのインポートとライブラリの配置・設定です。

ヘッダファイル

ソースコードと同じフォルダにヘッダファイル AquesTalk1-mac.h をコピーします。

C++アプリの場合は、AquesTalk1 の関数を呼び出すソースコードにインクルードします。

```
#include “AuesTalk1-mac.h”
```

Swift の場合は、ブリッジングヘッダーを用意し、その中でヘッダファイルをインポートします。

HelloAquesTalk-Bridging-Header.h

```
#import “AuesTalk1-mac.h”
```

Build Setting タブ → Objective-C Bridging Header :
HelloAquesTalk/HelloAquesTalk-Bridging-Header.h

ライブラリファイル

プロジェクトに libAquesTalk1-XX.dylib をコピーします(XX は声種)。コピー場所はプロジェクトディレクトリ(.xcodeproj が置かれているフォルダ)がおすすめ。

ビルド時に dylib を見つけるための設定
Build Settings タブ → Library Search Paths :
ビルド時の libAquesTalk1-XX.dylib のパスを指定（例：$(PROJECT_DIR) など）。

Build Phases タブ → Link Binary With Libraries :
libAquesTalk1-f1.dylib を指定。

.dylib は実行時にも必要です。実行時に libAquesTalk1-XX.dylib の配置場所は各種可能ですが、コマンドラインアプリの場合は実行モジュールと同じフォルダに、.app の場合は Frameworks フォルダ内に配置するのがおすすめです。

```
MyApp.app/
├── Contents/
    ├── Frameworks/
        └── libAquesTalk1-f1.dylib
```

実行時に dylib を見つけるための設定も必要です。

Build Settings タブ → Runpath Search Paths :
実行時の libAquesTalk1-XX.dylib のパスを指定します。
実行モジュールと同じフォルダの場合は、
@executable_path
Frameworks フォルダ内の場合は、
@executable_path/../Frameworks

Tips

■実行時に次のようなエラーが出る場合

```
dyld: Library not loaded: libAquesTalk-f1.dylib
```

指定したフォルダに.dylib があるか、また、下記の設定をチェックします。
　Build Settings → Runpath Search Paths に .dylib のパスを追加。
設定が正しくても、配置場所によっては sandobox で弾かれることがあります。

■ビルド時にアプリバンドル内に dylib を自動で入れる設定

Build Phases → Copy Files を追加。
Destination を Frameworks に設定。
libAquesTalk-XX.dylib を追加。

これでビルドされたとき
MyApp.app/Contents/Frameworks/ libAquesTalk-XX.dylib
に自動で配置されます。

■変数の型

AquesTalk_Synthe_Utf8()関数の引数や戻り値は C の変数型のため、Swift などで使用する場合は型変換が必要です。これはサンプルアプリを参考にしてください。

■AquesTalk_FreeWav()

AquesTalk_Synthe_Utf8()関数で返された音声データは、使用後に必ず AquesTalk_FreeWav()で解放してください。アプリ側で free() を使用して解放するなど異なる C ランタイムを使うとハングします。

■声種を切り替える

AquesTalk1 ライブラリは声種毎に異なる.dylib ファイルなので、アプリで動的に声種を切り替える場合は dlopen()を使って libAquesTalk-XX.dylib を動的にロードするようにします。

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。

AquesTalk_SetDevKey() をアプリケーションの起動初期（または、dlopen()でライブラリをロードした直後）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（または、dlopen()でライブラリをロードした直後）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。
通常は、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てください。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、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。

※AquesTalk_SetDevKey()/AquesTalk_SetUsrKey()の設定は永続化されないので、ライブラリをロードする都度設定する必要があります。

文書履歴

日付	版	更新内容
2007/01/07	1.0	新規作成 Win 版から加筆修正
2010/01/06	1.3	ライセンスキー不要版 HelloAqTk 説明追加
2025/04/20	2.0	Ver.2.0 用に全面改訂