株式会社 AQUEST
http://www.a-quest.com/

AquesTalk2　Win Manual

1. 概要

本文書は、規則音声合成ライブラリ「AquesTalk2 Win」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に関しての方法および注意点を示したものです。

AquesTalk2 は、簡単に小型機器への組み込みが出来る音声合成ミドルウェアです。
このライブラリを用いることで、簡単にテキスト情報を音声波形に変換出力することが可能になります。

AquesTalk2 には 2 種類のライブラリがあります。音声データをメモリ上に生成するものと、サウンドデバイスに出力する2種類があります。使用するアプリケーションに応じて選択してください。

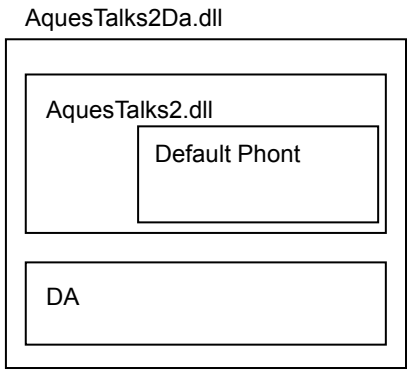
最も簡単な使用方法は、次の1行のコードで実現できます（VC++）。

1. 音声を生成して、サウンドデバイスに出力します

```
AquesTalk2Da_PlaySync ("こんにちわ。"); //< 引数に音声記号列の文字列を指定
```

2. ライブラリ構成

AquesTalk2 には、次の 2 種類のライブラリがあります。



1. **AquesTalk2.dll**
   音声記号列から音声データ(WAV フォーマット)を生成します。
   音声データをメモリ上に生成します。
   生成した音声データになんらかの処理を施す場合には、こちらを用います。

2. **AquesTalk2Da.dll**
   AquesTalk2.dll に、DA(サウンド出力機能)を含んだもので、音声記号列から音声を生成し、サウンドデバイスに出力します。
   AquesTalk2Da.dll の実行に際し AquesTalk2.dll は不要です。
   同期と非同期の 2 種類があります。同期タイプは発声を終了するまで関数から戻らないもので、非同期タイプは、発声の終了を待たずに関数から戻るもので、発声の終了はメッセージで通知することが可能です。

また、パッケージには、それぞれの DLL に対応した .lib, .h が含まれています。

2.1. Phont ファイル

パッケージには、phont ディレクトリに、声種を規定する phont ファイルがいくつか含まれています。なお、この中の aq_rm.phont は、DLL に内臓のデフォルト Phont と同じものです。
今後、公開される新しい Phont ファイルは、別途ダウンロードしてお使いください。

3. コンパイル･リンク

3.1. ヘッダ、ライブラリ

DLL を使用するには 対応するヘッダファイル(.h)をインクルードし、 リンク時に 対応する lib ファイルをリンクするか、LoadLibrary()などで実行時に動的にリンクすることが必要です。
各 DLL に対応するヘッダ等は下表を参照してください

DLL	ヘッダ	lib
AquesTalk2.dll	AquesTalk2.h	AquesTalk2.lib
AquesTalk2Da.dll	AquesTalk2Da.h	AquesTalk2Da.lib

3.2. 標準ライブラリ

本ライブラリは、実行時に標準ライブラリを別途必要とします(AquesTalk2.dll は、MSVCRT, KERNEL32。AquesTalk2Da は MSVCRT, KERNEL32,WINMM,USER32)。これらは通常 Windows のシステムディレクトリに含まれていますので、通常、実行時に用意する必要はありません。

ヒープメモリ処理のライブラリの関係上、AquesTalk2_Synthe()関数で返された音声データは、free()で解放せずに、AquesTalk2_FreeWav()を呼び出して解放してください。

ちなみに本ライブラリは、標準ライブラリを以下のように指定して VC++6.0 でビルドしています。

リリース版	マルチスレッド(DLL)

4. 関数 API

4. 1. AquesTalk2.dll

AquesTalk2_Synthe

AquesTalk2.h

説明 音声記号列から音声波形を生成します
生成した音声データは、使用後、呼び出し側で free()してください。

構文 unsigned char * **AquesTalk2_Synthe**(const char **koe*, int *iSpeed*, int **size, void *phontDat=0*)

引数

koe 音声記号列

iSpeed 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)

phontDat phont データの先頭アドレスを指定します。
この DLL のデフォルト Phont を用いるときは0を指定します。

戻り値 WAV フォーマットのデータ(内部で領域確保、解放は呼び出し側で AquesTalk2_FreeWave()で行う)の先頭アドレスを返す。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。

SetSpeed AquesTalk2_FreeWave

AquesTalk2.h

説明 音声データの領域を開放

構文 void **AquesTalk2_FreeWave** (unsigned char *wav)

引数 なし

wav WAV フォーマットのデータ(AquesTalk2_Synthe()で生成した音声データ)

戻り値 なし

4.2. AquesTalk2Da.dll

AquesTalk2Da_PlaySync

AquesTalk2Da.h

説明 同期タイプの音声合成。発声が終了するまで戻らない。

構文 int **AquesTalk2Da_PlaySync**(const char **koe*, int *iSpeed=100 , void *phontDat=0*)

引数

koe 音声記号列

iSpeed 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

phontDat phont データの先頭アドレスを指定します。

この DLL のデフォルト Phont を用いるときは0を指定します。

戻り値 0:正常終了 それ以外:エラー

AquesTalk2Da_Create

AquesTalk2Da.h

説明 音声合成エンジンのインスタンスを生成(非同期タイプ)

構文 H_AQTKDA **AquesTalk2Da_Create**()

引数

戻り値 音声合成エンジンのハンドル

AquesTalk2Da_Release

AquesTalk2Da.h

説明 音声合成エンジンのインスタンスを解放(非同期タイプ)

発声終了前にこの関数でインスタンス解放すると、その時点で発声が終了してしまうので注意

構文 void **AquesTalk2Da_Release**(H_AQTKDA *hMe*)

引数

hMe 音声合成エンジンのハンドル

戻り値 なし

AquesTalk2Da_Play

AquesTalk2Da.h

説明 非同期タイプの音声合成。発声終了を待たずに戻る。

発声終了時に hWnd に指定したウィンドウにメッセージが送出(post)される。

再生終了前に AquesTalk2Da_Play()を再度呼び出すことで、連続再生可能。また、このとき、hWnd 等を変更して異なるメッセージを設定することも可能。

※phont データは、発声が完了するまで開放してはいけません。

構文 int **AquesTalk2Da_Play**(H_AQTKDA *hMe,* const char *koe, int *iSpeed=100*, void *phontDat=0, HWND *hWnd*=0, unsigned long *msg*=0, unsigned long dwUser=0)

引数

hMe 音声合成エンジンのハンドル

koe 音声記号列

iSpeed 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

phontDat phont データの先頭アドレスを指定します。

この DLL のデフォルト Phont を用いるときは0を指定します。

hWnd	発声終了時のメッセージの送り先 Window ハンドル（NULL を指定すると終了メッセージは送られない）
msg	発声終了時のメッセージ ID を指定する。hWnd=NULL の時は無効
dwUser	任意。発生終了時のメッセージの lParam(第 2 引数)に渡される
戻り値	0：正常終了 それ以外：エラー

AquesTalk2Da_Stop

AquesTalk2Da.h

説明	発声の停止。Play()で発声中に使用する。 Stop()によって発声が終了した場合も、Play()で *hWnd* が指定されていたならメッセージが送出される。
構文	void **AquesTalk2Da_Stop**(H_AQTKDA *hMe*)
引数	
hMe	音声合成エンジンのハンドル
戻り値	なし

AquesTalk2Da_IsPlay

AquesTalk2Da.h

説明	再生中か否か
構文	int **AquesTalk2Da_IsPlay**(H_AQTKDA *hMe*)
引数	
hMe	音声合成エンジンのハンドル
戻り値	1:再生中 0:再生中でない

5. 音声データ形式

本ライブラリで生成する音声データは、次の形式となります。
AquesTalk2_Synthe()で生成する音声データには、先頭部に WAV ヘッダが付与されています。
ストレート PCM データが必要な場合は、別途ヘッダを除いて使用してください。
また、AquesTalk2Da.dll では、サウンドドライバが以下の形式の音声を再生できる必要があります(基本的な形式ですので、ほとんどのパソコンで問題なく再生できると思います)

サンプリング周波数	8KHz
量子化 bit 数	16bit
チャンネル数	モノラル
エンコード	リニア PCM
フォーマット	WAV 形式

6. サンプルコード

次に示すコードは、音声記号列から音声データを生成し、WAV ファイルとして出力する最も単純なプログラムです(HelloTalk.cpp)。
12行目の"こんにちわ。" の部分を、任意の音声記号列に変更することで、異なるメッセーを生成可能です。
なお、このプログラムで出力した WAV ファイルは、メディアプレイヤーなどで再生することができます。

```
#include <stdio.h>
#include <AquesTalk2.h>     // AquestTalk クラスのヘッダ
```

```
int main(int ac, char **av)
{
    int size;


    // メモリ上に音声データを生成
    unsigned char *wav = AquesTalk2_Synthe("こんにちわ。", 100, &size, 0);
    if(wav==0) {
        fprintf(stderr, "ERR %d", size); // エラー時は size にエラーコードが返る
        return -1;
    }

    //   ルートディレクトリに生成した音声データを保存
    FILE *fp = fopen("¥¥ZZZ.wav", "wb");
    fwrite(wav, 1, size, fp);
    fclose(fp);

    // Synthe()で生成した音声データは、使用後に呼び出し側で解放する
    AquesTalk2_FreeWave (wav);

    return 0;
}
```

次に示すコードは、先のコードに外部の Phont ファイルを指定できるようにしたものです。
Phont ファイルをバイナリで読み込んで、先頭アドレスを合成時に指定しています。

```
#include <stdio.h>
#include <memory.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <AquesTalk2.h>     // AquestTalk クラスのヘッダ

void * file_load(const char * file, int * psize);

int main(int ac, char **av)
{
    int size;

    // Phont ファイルの読み込み
    int size;
    void *pPhont = file_load("¥¥aq_f1b.phont", &size); // Phont ファイルをここで指定
    if(pPhont ==0) return-2;

    // メモリ上に音声データを生成
    unsigned char *wav = AquesTalk2_Synthe("こんにちわ。", 100, &size, pPhont);
    if(wav==0) {
        fprintf(stderr, "ERR %d", size); // エラー時は size にエラーコードが返る
        return -1;
    }
    // Phont データの開放（音声合成が終わったら開放できる）
    free(pPhont);

    //   ルートディレクトリに生成した音声データを保存
    FILE *fp = fopen("¥¥ZZZ.wav", "wb");
    fwrite(wav, 1, size, fp);
    fclose(fp);

    // Synthe()で生成した音声データは、使用後に呼び出し側で解放する
    AquesTalk2_FreeWave (wav);

    return 0;
}

// ファイルの読み込み
void * file_load(const char * file, int * psize)
{
    FILE    *fp;
    char    *data;
```

```
    struct _stat     st;

    *psize = 0;

    if( _stat(file, &st)!=0) return NULL;

    if((data=(char *)malloc(st.st_size))==NULL){
        fprintf(stderr,"can not alloc memory(file_load)¥n");
        return NULL;
    }

    if((fp=fopen(file,"rb"))==NULL) {
        free(data);
        perror(file);
        return NULL;
    }
    if(fread(data, 1, st.st_size, fp)<(unsigned)st.st_size) {
        fprintf(stderr,"can not read data (file_load)¥n");
        free(data);
        fclose(fp);
        return NULL;
    }
    fclose(fp);
    *psize = st.st_size;
    return data;

}
```

非同期に音声出力を行う、再生を停止する、発声速度を変更するなどの、より高度なプログラミング方法は、AquesTalk ライブラリ（この AquesTalk2 パッケージではありません）付属の MFC アプリ AqTkApp のソースコードを参考にしてください。
VC++以外の環境や、他の言語（C#,VB,PHP など）での使用方法はここでは示しませんが、呼び出して使用することが出来ると思います。（ポインタを返す関数は、VB で使う場合にはラッパーが必要になるかもしれません）

7. 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
102	音声記号列に未定義の読み記号が指定された
103	韻律データの時間長がマイナスなっている
104	内部エラー(未定義の区切りコード検出)
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
109	WAVE 再生ができない（サウンドドライバ関連の問題）
110	WAVE 再生ができない（サウンドドライバ関連の問題　非同期再生）
111	発声すべきデータがない
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	音声記号列が長い（内部バッファオーバー1）
203	ヒープメモリ不足
204	音声記号列が長い（内部バッファオーバー1）
1000 - 1008	Phont データが正しくない

8. 履歴

日付	版	変更箇所	更新内容	更新者
2009/12/26	1.0	新規作成		N.Y
2011/01/28	2.1		パッケージ同梱用に修正	N.Y