Ver. 1.1

2011/6

USB メモリライセンス認証　API

有限会社リビッグ

〒233-0002 横浜市港南区上大岡西 1-12-2

Tel: 045-843-7122 Fax: 045-843-7142

http://www.ribig.co.jp

USB メモリライセンス認証　API

USB メモリライセンス認証を行う API を含む DLL です。API を利用することでアプリケーションプログラムから容易に USB メモリライセンスの確認が行えます。

アプリケーションプログラムから DLL を利用するには、DLL ファイルを LoadLibrary でロードして関数アドレスを取得から呼び出します。関数は名前ではなく序数になっています。ライセンス認証 DLL のインポートライブラリは提供していません

1.　DLL関数プロトタイプ

DLLは４つの関数を公開します。

```
//DLL初期化
typedef int      (__stdcall *USBKeyInit)();

//DLLの認証
typedef void     (__stdcall *USBKeyDllAuth)(unsigned long*);

//USBメモリ認証(UNICODE版)
typedef float (__stdcall *USBKeyQuery)(WCHAR*, WCHAR*);

//USBメモリ認証　ANSII版
typedef float (__stdcall *USBKeyQueryA)(char*, char*);
```

2　関数アドレス取得

DLL ファイルを LoadLibrary でロード後、序数で関数アドレスを取得してください。

```
//DLL初期化
USBKeyInit　initUSBKey = (USBKeyInit)GetProcAddress(hMod, (LPCSTR)数値);

//DLLの認証
USBKeyDllAuth auth = (USBKeyDllAuth)GetProcAddress(hMod, (LPCSTR) 数値);

//USBメモリ認証
USBKeyQuery keyQuery=(USBKeyQuery)GetProcAddress(hMod, (LPCSTR) 数値);


//USBメモリ認証
USBKeyQueryA keyQueryA=(USBKeyQueryA)GetProcAddress(hMod,(LPCSTR) 数値);
```

*数値は固定されていません。評価版の数値は api_test フォルダ内のサンプルプログラムを参照ください。

3. DLL 名

APIを含むDLLのファイル名は任意です。LoadLibraryで読み込みますので、拡張子が DLL である必要もありません。

配布ディスクの api_testフォルダにサンプルプログラムがあります。APIを利用したコーディング例は api_test.cpp をご参照ください

4. DLL初期化

int USBKeyInit(void)	
LoadLibraryでDLLをロード後、最初にこの関数を呼び出します。この関数を呼び出す前に、他の関数呼び出しを行うとエラーになります。	
引数	無し
戻り値	0：成功 0以外：失敗

5. DLLの認証

void USBKeyDllAuth(unsigned long *)	
ロードしたDLLが間違えなく本物のUSBキーライセンス認証DLLであることを認証する関数です。	
引数	IN: ２つの unsigned long 値を含むバッファへのポインタ OUT: 暗号化された２つのunsigned long値を含むバッファ
戻り値	無し

認証は、呼び出し側と呼び出された側がそれぞれ共通鍵を用いて同じデータを暗号化/復号化した結果の比較で行います。

1. DLLテンプレートファイルから実DLL生成時に４つの暗号鍵を指定します。この暗号鍵は、暗号化されて実DLLに埋め込まれます。
2. この関数が呼び出されると、埋め込まれた暗号鍵で引数を暗号化します。
3. 関数の呼び出し側は、同じ暗号鍵を用いて
   ① 引数として渡した値を暗号化して、関数から戻される引数と比較
   ② 関数から戻される引数を復号化して、渡した値と比較
   一致したら認証成功とします。

実DLL生成時に指定した４つの暗号鍵については DLL生成担当者にお尋ねください。評価版DLLは 1111 2222 3333 4444 で生成されています。

暗号化/復号化は XTEAで行います。XTEA関数を含むヘッダファイル xtea.h をソースにインクルードしてください。

```
暗号化　short MxApp_Encrypt(unsigned long *DataBlock, unsigned long *Key);
復号化　short MxApp_Decrypt(unsigned long *DataBlock, unsigned long *Key);
```

```
#include “mxtea.h”

unsigned long key[4];
key[0]=1111; key[1]=2222; key[2]=2222; key[3]=3333;

unsigned long data[2], data1[2];
data[0] = data1[0] = xxxxx; data[1] = data1[1] = yyyyy;

USBKeyAllAuth( data );

MxApp_Decrypt(data, key );

If( data[0]==data1[0] && data[1] == data1[1] )
        // authenticated successfully
```

6. USBメモリ認証

float USBKeyQuery(WCHAR*, WCHAR*)　// UNICODE版 float USBKeyQueryA(char*, char*)　// ANSI版	
USBメモリが接続されていて、正規のライセンス情報が書き込まれているかどうかを確認します。	
引数	第一引数: MAX_PATH以上の長さをもったワイド文字列へのポインタ、又は NULL 第二引数: 3 以上の長さをもったワイド文字列へのポインタ、またはNULL
戻り値	成功: 182で割切れる値 失敗 182で割り切れない値

成功すると、第二引数に確認したライセンス情報を保存している USB メモリのドライブがセットされます。第一引数には空文字列が戻されます。

関数は　182 で割り切れる数を戻します。

成功テスト例

```
float r = USBKeyQuery(NULL, driveLetter )
if( !((long)r % 182) )
        // 成功
else
        // 失敗
```