



voiXtreme

**Spécifications pour l'utilisation de la
bibliothèque de la voix**

voiXtreme_dll_fr_v100.doc

20080310 Version 1.0 Fr	
-------------------------	--

Table de matières

INTRODUCTION	3
RESUME	3
PRÉSENTATION DU SYSTÈME	3
EXEMPLES D'UTILISATION	3
DÉMARRAGE, FONCTIONS DÉFINIES ET SYNTAXE	4
A.- TTS	5
B.- ASR	5
C.- ACTIVATION DE LA PARTIE DE LA GRAMMAIRE QUI CONTIENT DES DIGITS UNIQUEMENT	5
SYNTAXE DES FONCTIONS ET PROTOTYPES	5
A.- DEMARRAGE, ARRET ET CONFIGURATION	5
B.- TTS	6
C.- ASR	6
LICENCES D'UTILISATION	7



Introduction

Résumé

Ce document présente le mode d'utilisation du système de reconnaissance vocale (*ASR – Automatic Speech Recognition*) et de la synthèse vocale (*TTS – Text to Speech*) à travers la librairie `voixtreme.dll`.

Ce texte, dirigé au programmeur, est une version préliminaire qui présente l'ensemble minimum d'options nécessaires pour une utilisation initiale. Notre but est d'inclure cette version dans l'installation des composants à fin de recevoir des suggestions des programmeurs et utilisateurs.

Présentation du système

On appellera "voiXtreme", "SofToGo Voice Library" ou simplement "VoiceLib" à l'ensemble de deux systèmes puissants qui sont en rapport avec la parole et les systèmes électroniques -*ASR (Automatic Speech Recognition)* et *TTS (Text to Speech)*- avec la possibilité d'accéder à leurs méthodes à travers une dll unique développée pour Windows Mobile.

En lignes générales, ASR écoute une allocution déterminée, il l'analyse selon une grammaire prédéterminée et retourne une chaîne de texte -appartenant à cette grammaire- la plus proche possible à l'allocution prononcée.

A l'inverse, le TTS permet de convertir en sons des ordres écrits, de sorte qu'il est possible d'écouter un texte qui se trouve sous la forme de chaîne de caractères.

On présente ici un manuel du programmeur qui permettra d'accéder aux fonctionnalités de `voixtreme.dll`. Chaque composant est présenté pour le mettre en discussion, de sorte qu'il est possible qu'il y ait des modifications entre les versions successives. Ce document sera mis à jour comme conséquence de ces modifications.

Exemples d'utilisation

Avec l'installation on présente une application, `voixtreme_cpp_test.exe` (qui peut être lancée depuis "Start/voiXtreme cpp Test" ou depuis "Start/Programs/voiXtreme cpp Test"), développée en C++ et compilée avec Microsoft Visual Studio 2005 (Professional Edition) Version 8.0.50727.762 (SP.050727-7600).

Il est possible d'obtenir le code source de cette application et celui d'autre application similaire en Visual Basic depuis le site <http://www.softogo.com>.

Le programmeur devrait connaître la grammaire installée, que pour cet exemple est la suivante :

```
!grammar Digit;
!start <Speech>;

<Speech>: !repeat(<Digit>,1,*) | <digitex> | <commande> ;

<Digit>: 0 !id(0) |
         1 !id(1) |
         2 !id(2) |
         3 !id(3) |
         4 !id(4) |
         5 !id(5) |
         6 !id(6) |
         7 !id(7) |
         8 !id(8) |
         9 !id(9) ;

<digitex>: annuler |
          répéter ;

<commande> : accepter |
            commencer |
            finir ;
```

Démarrage, fonctions définies et syntaxe

Après charger la dll avec la fonction LoadLibrary, il faut démarrer la librairie avec la fonction StgVoiceInit.

Avant de finir d'utiliser la *Voice*, normalement, au moment de sortir de l'application, il faut arrêter la librairie à travers la fonction StgVoiceDeInit.

```
// Démarrer
m_hStgVoice = LoadLibrary( _T("stgvoice.dll") ) ;
m_StgVoiceSetWindow = (StgVoiceSetWindowFn)GetProcAddress(
m_hStgVoice, _T("StgVoiceSetWindow"));
m_StgVoiceSetWindow( m_hWnd ) ;
m_StgVoiceInit = (StgVoiceInitFn)GetProcAddress( m_hStgVoice,
_T("StgVoiceInit"));
m_StgVoiceInit( ) ;
m_StgVoiceSay = (StgVoiceSayFn)GetProcAddress( m_hStgVoice,
_T("StgVoiceSay"));
m_StgVoiceListen = (StgVoiceListenFn)GetProcAddress( m_hStgVoice,
_T("StgVoiceListen"));
m_StgVoiceActivateRule = (StgVoiceActivateRuleFn)GetProcAddress(
m_hStgVoice, _T("StgVoiceActivateRule"));
m_StgVoiceDeactivateRule = (StgVoiceDeactivateRuleFn)GetProcAddress(
m_hStgVoice, _T("StgVoiceDeactivateRule"));

// Arrêter
m_StgVoiceDeInit = (StgVoiceDeInitFn)GetProcAddress( m_hStgVoice,
_T("StgVoiceDeInit"));
m_StgVoiceDeInit( ) ;
```



Des exemples des appels

a.- TTS

```
m_StgVoiceSay( _T("Bonjour, bienvenu à cette application.") ) ;
```

b.- ASR

Lorsqu'on appelle :

```
m_StgVoiceListen( ) ;
```

on attend le message WMU_RESULT, qui est envoyé à la fenêtre (configurée à travers un appel à StgVoiceSetWindow) pour recevoir le string résultant du ASR.

```
#define WMU_RESULT (WM_USER + 4)
```

Spécification de la fonction appel quand le message WMU_RESULT ON_MESSAGE(WMU_RESULT, &CdlltestDlg::OnResultGiven) arrive.

Quand le message arrive, on appelle :

```
LRESULT CdlltestDlg::OnResultGiven(WPARAM wParam, LPARAM lParam)
{
    wchar_t * strResult = (wchar_t *) lParam ;
    SetDlgItemText( IDC_EDIT1, strResult ) ;
    return 0 ;
}
```

qui configure le texte sur le champ IDC_EDIT1.

c.- Activation de la partie de la grammaire qui contient des digits uniquement

```
m_StgVoiceActivateRule( _T("<Digit>") ) ;
m_StgVoiceDeactivateRule( _T("<commande>") ) ;
m_StgVoiceDeactivateRule( _T("<digitex>") ) ;
```

Syntaxe des fonctions et prototypes

a.- Démarrage, arrêt et configuration

```
STDAPI StgVoiceInit( void ) ;
```

Démarre la librairie du Voice (ASR+TTS). Il est nécessaire d'appeler cette fonction avant d'appeler n'importe quelle autre fonction.

```
STDAPI StgVoiceDeInit( void );
```

Complément de StgVoiceInit. Il faut appeler cette fonction quand on finit d'utiliser le Voice.

```
STDAPI StgVoiceSetWindow( HWND p_hWnd ) ;
```

Configure la fenêtre qui recevra le message WMU_RESULT avec le résultat de la reconnaissance.



STDAPI `StgVoiceSetDebuggingMode(bool p_bShowMessageBoxes) ;`
Permet l'activation du mode de debugging. Dans ce mode, chaque erreur d'exécution lance un message de texte communiquant l'erreur. Il est désactivé par défaut.

STDAPI `StgVoiceActivateRule(wchar_t * p_strRule) ;`
Active une règle de la grammaire. Cela *permet* que la fonction `Listen` retourne une chaîne de caractères générée avec cette règle.

STDAPI `StgVoiceDeactivateRule(wchar_t * p_strRule) ;`
Désactive une règle de la grammaire. Cela *empêche* que la fonction `Listen` puisse retourner une chaîne de caractères générée avec cette règle.

b.- TTS

STDAPI `StgVoiceSay(wchar_t * p_strToSay) ;`
Lit la chaîne de caractères passée en paramètre et retourne un code d'erreur. 0 (zéro) si ok.

STDAPI `StgVoiceSpell(wchar_t * p_strToSpell) ;`
Épelle la chaîne de caractères numérique passée en paramètre et retourne un code d'erreur. 0 (zéro) si ok.

STDAPI `StgVoiceIsSpeaking(void) ;`
Retourne une valeur booléenne indiquant si le TTS est actif et en train de prononcer ou épeler un texte.

STDAPI `StgVoiceStopSpeaking(void) ;`
Termine abruptement la prononciation d'une phrase.

STDAPI `StgVoiceSaySync(wchar_t * p_strToSay) ;`
Lit la chaîne de caractères passée en paramètre et retourne un code d'erreur. 0 (zéro) si ok. Retourne le control uniquement après avoir prononcé toute la chaîne de texte passée en paramètre.

STDAPI `StgVoiceSpellSync(wchar_t * p_strToSpell) ;`
Épelle la chaîne de caractères numérique passée en paramètre et retourne un code d'erreur. 0 (zéro) si ok. Retourne le contrôle uniquement après avoir prononcé toute la chaîne de texte passée en paramètre.

c.- ASR

STDAPI `StgVoiceListen(void) ;`
Lance le reconaisseur de la voix de l'utilisateur. La fonction est asynchrone, c'est-à-dire que le système retourne immédiatement le contrôle et permet de suivre avec l'exécution des lignes de code suivantes. Le programmeur doit programmer le handler pour le traitement du message `WMU_RESULT`, c'est le moment où le ASR retourne la chaîne de texte reconnue. En option, l'utilisateur peut attendre la fin de la reconnaissance en évaluant la fonction `StgVoiceIsListening`, jusqu'à ce qu'elle retourne `false`. A ce moment, la chaîne résultante peut être obtenue à travers la fonction `StgVoiceGetString`.

STDAPI `StgVoiceListenSync(wchar_t * p_strResult) ;`



Retourne une chaîne de caractères, résultant de la reconnaissance de la voix de l'utilisateur. La fonction est synchrone, c'est-à-dire que le système doit attendre le résultat de cette fonction pour continuer avec l'exécution des lignes de code suivantes.

```
STDAPI StgVoiceIsListening( void )
```

Retourne une valeur booléenne indiquant si le ASR est actif et en train de recevoir des inputs du système d'audio.

```
STDAPI StgVoiceGetResultString( wchar_t * p_strResult ) ;
```

Retourne la dernière chaîne de texte reconnue par l'ASR.

```
STDAPI StgVoiceStopListening( void ) ;
```

Termine abruptement le reconnaisseur de la parole sans retourner un résultat.

Les paramètres du système vocal sont préconfigurés avec des options par défaut. Les méthodes et propriétés qui permettent une configuration plus détaillée seront ajoutées dans des versions ultérieures.

Licences d'utilisation

Pendant que la librairie n'est pas enregistrée, quand on lance la fonction `Init` un message indiquant la manque de registre apparaît. Dans cet état, le ASR et le TTS modifient tous les caractères A et les digits 0 (zéro), et retournent ou lisent E et 5 (cinq) respectivement.

La librairie se registre depuis l'accès direct `voiXtremeLic` (`STGVoiceLic.exe`), directement depuis le terminal Windows Mobile ou à travers l'accès direct sur le PC: `Start/Programs/WireLess Utilities/voiXtreme Libraries/Es/Licencer` (les PDA).

Pour avoir plus d'informations sur la registration des librairies, renseignez-vous la documentation "Licences Laissez-moi (pdf)", sur le même répertoire.