



voiXtreme

**Specifications for using the voice
library**

voiXtreme_dll_en_v100.doc

| | |
|-------------------------|--|
| 20080310 Version 1.0 En | |
|-------------------------|--|

Index

| | |
|--|----------|
| INTRO | 3 |
| OVERVIEW..... | 3 |
| SYSTEM DESCRIPTION | 3 |
| USE EXAMPLE | 3 |
| START, DEFINED FUNCTIONS AND SYNTAX | 4 |
| A.- TTS | 5 |
| B.- ASR..... | 5 |
| C.- ACTIVATING THE PART OF THE GRAMMAR CONTAINING ONLY DIGITS..... | 5 |
| FUNCTION SYNTAX AND PROTOTYPES | 5 |
| A.- START, SHUTDOWN AND CONFIGURATION | 5 |
| B.- TTS..... | 6 |
| C.- ASR..... | 6 |
| LICENSE | 7 |



Intro

Overview

This document describes how to use the speech recognition system (*ASR – Automatic Speech Recognition*) and the speech synthesis system (*TTS – Text to Speech*), through `voixtreme.dll` library.

This document, intended for programmers, describes a preliminary version that presents only those features necessary for a basic use. Our purpose is to include it on our packages in order to receive programmer and user feedback.

System description

We will refer as “*voiXtreme*”, “*SofToGo Voice Library*” or just “*VoiceLib*” to the set of two powerful systems related to speech and electronic systems -*ASR (Automatic Speech Recognition)* and *TTS (Text to Speech)* - and the possibility to take advantage of them using a unique dll developed for Windows Mobile.

Basically, ASR listens the user's speech, it process it following a predefined grammar and returns a string -belonging to that grammar- as similar as possible to the user's speech.

TTS performs the opposite process: It acts as a "reader", i.e., it can turn a character chain into articulated sounds.

This programmer manual is an introduction to `voixtreme.dll` features. Each one of them is presented and left under discussion; therefore, changes may occur between versions. Those changes will be reflected on an update of this document.

Use example

We also submit an application (`voixtreme_cpp_test.exe`, which can be launched from “Start/voixtreme cpp Test” or from “Start/Programs/voixtreme cpp Test”) developed on C++, compiled with Microsoft Visual Studio 2005 (Professional Edition) Version 8.0.50727.762 (SP.050727-7600).

It is possible to get the source code of this application and of another similar application on Visual Basic from the website <http://www.softogo.com>.



The programmer should know the installed grammar. For our example, it should be as follows:

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

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

<digits>: 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) ;

<commande> : ok      |
            cancel   |
            begin    |
            end ;
```

Start, defined functions and syntax

After loading the dll using `LoadLibrary` function, you must start the library using `StgVoiceInit` function.

Before finishing using *Voice*, when exiting the application, you must shutdown the library using `StgVoiceDeInit` function.

```
// Start
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"));

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



Calling examples

a.- TTS

```
m_StgVoiceSay( _T("Hello, welcome to this application.") ) ;
```

b.- ASR

When calling:

```
m_StgVoiceListen( ) ;
```

the message WMU_RESULT is expected, and is sent to a window (configured calling StgVoiceSetWindow) to receive the string resulting of ASR.

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

Specification of the function called when the message WMU_RESULT ON_MESSAGE(WMU_RESULT, &CdlltestDlg::OnResultGiven) arrives.

When the message arrives, the application calls this function:

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

which sets the text on IDC_EDIT1 field.

c.- Activating the part of the grammar containing only digits

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

Function syntax and prototypes

a.- Start, shutdown and configuration

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

Starts the "Voice" (ASR+TTS) library. It is necessary to call this function before calling any other function.

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

StgVoiceInit counterpart. You must call this function when you end using "Voice".

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

Sets the window that will receive the message WMU_RESULT with the result of the recognition.



STDAPI `StgVoiceSetDebuggingMode(bool p_bShowMessageBoxes) ;`
Allows to activate debugging mode. On this mode each execution error launches a text message communicating the error. Disabled by default.

STDAPI `StgVoiceActivateRule(wchar_t * p_strRule) ;`
Activates a grammar rule. This allows the "Listen" function to return a string generated according to this rule.

STDAPI `StgVoiceDeactivateRule(wchar_t * p_strRule) ;`
Deactivates a grammar rule. This disables the "Listen" function to return a string generated according to this rule.

b.- TTS

STDAPI `StgVoiceSay(wchar_t * p_strToSay) ;`
Reads the string passed as parameter and returns an error code. 0 if ok.

STDAPI `StgVoiceSpell(wchar_t * p_strToSpell) ;`
Spells the numeric string passed as parameter and returns an error code. 0 if ok.

STDAPI `StgVoiceIsSpeaking(void) ;`
Returns a Boolean value indicating if TTS is active and reading or spelling text.

STDAPI `StgVoiceStopSpeaking(void) ;`
Ends abruptly the reading of a phrase.

STDAPI `StgVoiceSaySync(wchar_t * p_strToSay) ;`
Reads the string passed as parameter and returns an error code, 0 if ok. The system waits the reading of the whole string passed as parameter to be finished in order to continue with the next instruction.

STDAPI `StgVoiceSpellSync(wchar_t * p_strToSpell) ;`
Spells the numeric string passed as parameter and returns an error code, 0 if ok. The system waits the spelling of the whole string passed as parameter to be finished in order to continue with the next instruction.

c.- ASR

STDAPI `StgVoiceListen(void) ;`
Launches the user's speech recognizer. The function is asynchronous, i.e., control returns immediately and execution continues with the next instructions. The programmer must configure the handler to treat WMU_RESULT message, that's when the ASR returns the recognized string. The user may wait for the end of the recognition, while testing `StgVoiceIsListening`, until it returns `false`. That's when the resulting string can be obtained using `StgVoiceGetResultString` function.

STDAPI `StgVoiceListenSync(wchar_t * p_strResult) ;`
Returns a string resulting from the user's speech recognition. The function is synchronous, i.e., the system waits the result of this function in order to continue with the next instructions.



STDAPI `StgVoiceIsListening(void)`

Returns a Boolean value indicating if the ASR is active receiving audio input.

STDAPI `StgVoiceGetResultString(wchar_t * p_strResult) ;`

Returns the last string recognized by ASR.

STDAPI `StgVoiceStopListening(void) ;`

Ends abruptly the speech recognizer, without returning a result.

Vocal system parameters are predefined with default options. Methods and properties that will allow more detailed configuration will be added in later versions.

License

While the library isn't registered, when launching `Init` function, a message will appear, notifying the lack of registration. On this state, the ASR and the TTS modify all A characters and all 0 (zero) digits, returning or reading them as E and 5 (five) respectively.

The library can be registered from the direct access `voiXtremeLic` (`STGVoiceLic.exe`) from the Windows Mobile terminal or from the direct access on the PC: `Start/Programs/WireLess Utilities/voiXtreme Libraries/Es/License` (the PDA).

For more information on how to register the libraries, read "License ReadMe (pdf)", on the same directory.