



**Specifications for using voice picking  
on FlexBrowser.**

**The ActiveX voiXtreme.**

wfb\_voice\_en.doc

20080226 Version 1.0 EN	
-------------------------	--

## Introduction

Overview.

This document describes how to use the speech recognition system (*ASR – Automatic Speech Recognition*) and the speech synthesis system (*TTS – Text to Speech*) on a web application launched through *FlexBrowser*.

This text is intended to web application programmers and presents at the same time: an introduction, the syntax for the set of functions and examples.

## System description

We will refer as “*FlexBrowser Voice*” or just “*Voice*” to the set of two powerful systems related to speech and electronic systems - *ASR (Automatic Speech Recognition)* and *TTS (Text to Speech)* - and *FlexBrowser*.

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.

In both cases, the whole process is performed on the terminal, so there's no need of any connection to the server.

*FlexBrowser* uses both systems through an ActiveX control of its own called *voiXtreme*, which will be described further. For an HTML page to be able to use *Voice* it must follow the detailed specifications.

## voiXtreme

*voiXtreme.ocx* is the name of the ActiveX, installed separately from *FlexBrowser*, which allows using *Voice*. There's a different version for each language and, on each case, the ocx is installed together with the set of necessary data and libraries.

Due to the size of *voiXtreme* components and the fact that Windows Mobile 2005 systems keep their file system on persistent memory, the *Voice* is installed as warm and coldboot resistant, but not cleanboot resistant.

## Defined functions, parameters, returned values and syntax

The object that will be instantiated has the CLSID `5FF0CA2F-A640-4F62-8EDF-C97F30B562FB`, and exposes the following methods:

```
int StgVoiceInit( ) ;
```

Starts the "Voice" (ASR+TTS) library. It is necessary to call this function before calling any other function. Once started, remains active for all pages, i.e. it is not necessary to restart it.

```
int StgVoiceDeInit( ) ;
```

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

```
int StgVoiceSay( strToSay ) ;
```

Reads the string passed as parameter and returns an error code. 0 if ok.

```
int StgVoiceListen( ) ;
```

Launches the user's speech recognizer. The function is asynchronous, i.e., control returns immediately and execution continues with the next instructions. The user must 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.

```
str StgVoiceListenSync( ) ;
```

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.

```
int StgVoiceActivateRule( strRule ) ;
```

Activates a grammar rule. This allows the "Listen" function to return a string generated according to this rule.

```
int StgVoiceDeactivateRule( strRule ) ;
```

Deactivates a grammar rule. This *disables* the "Listen" function to return a string generated according to this rule.

```
int StgVoiceSetDebuggingMode(ULONG p_bShowMessageBoxes) ;
```

Allows to activate debugging mode. On this mode each execution error launches a text message communicating the error. Disabled by default.

```
int StgVoiceIsSpeaking( ) ;
```

Returns a Boolean value indicating if TTS is active and reading text.

```
str StgVoiceGetResultString( ) ;
```

Returns the last string recognized by ASR.

```
int StgVoiceIsListening( ) ;  
Returns a Boolean value indicating if the ASR is active receiving  
audio input.
```

```
int StgVoiceSpell( strToSpell ) ;  
Spells the numeric string passed as parameter and returns an error  
code. 0 if ok.
```

```
int StgVoiceStopListening( ) ;  
Ends abruptly the speech recognizer, without returning a result.
```

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

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

## Use example

On the following example we will instantiate an object as `VoiceObj`, which allows accessing the vocal system.

voixtremel.htm file

```
<HTML>
<HEAD>
<TITLE>Test page for Voice Object voiXtreme in FlexTools</TITLE>
<META http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<META NAME="MobileOptimized" content="240">
<SCRIPT LANGUAGE="javascript">
  function Init()
  {
    VoiceObj.StgVoicedInit() ;
    document.forms[0][0].disabled = true ;
    document.forms[0][1].disabled = false ;
  }
  function DeInit()
  {
    VoiceObj.StgVoiceDeInit() ;
    document.forms[0][0].disabled = false ;
    document.forms[0][1].disabled = true ;
  }
</SCRIPT>
</HEAD>

<BODY>
<OBJECT ID="VoiceObj" CLASSID="CLSID:5FF0CA2F-A640-4F62-8EDF-C97F30B562FB">
  <PARAM NAME="_Version" VALUE="65536">
</OBJECT>
  <P>Demo Voice with FlexTools - voiXtreme</P>
  <FORM id="form1">
    <P><INPUT type="button" id="init" name="Init" onclick="javascript:Init();"
value="Init"></P>
    <P><INPUT type="button" id="deinit" name="DeInit"
onclick="javascript:DeInit();" value="DeInit"></P>
  </FORM>
  <A HREF="voixtreme2.htm">Go</A>
</BODY>
</HTML>
```

voixtreme2.htm file

```
<HTML>
<HEAD>
<TITLE>Test page for Voice Object voiXtreme in FlexTools</TITLE>
<META http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<META NAME="MobileOptimized" content="240">
<SCRIPT LANGUAGE="javascript">
    function OnSubmit()
    {
        VoiceObj.StgVoiceSpellSync( document.forms[0][0].value ) ;
        VoiceObj.StgVoiceSaySync( "Ok, cancel or end?" ) ;

        VoiceObj.StgVoiceActivateRule("<commande>") ;
        VoiceObj.StgVoiceDeactivateRule("<digits>") ;
        VoiceObj.StgVoiceListenSync() ;
        commandsaid = VoiceObj.StgVoiceGetResultString() ;
        if ( commandsaid == "Ok" )
        {
            document.forms[0].submit();
        }
        else
        {
            document.forms[0][0].value = VoiceObj.StgVoiceGetResultString() ;
            if ( commandsaid != "end" )
            {
                Listen() ;
            }
            else
            {
                VoiceObj.StgVoiceSaySync( "Thank you. See you soon." ) ;
            }
        }
    }
    function Listen()
    {
        VoiceObj.StgVoiceSaySync("quantity") ;
        VoiceObj.StgVoiceActivateRule("<digits>") ;
        VoiceObj.StgVoiceDeactivateRule("<commande>") ;
        VoiceObj.StgVoiceListenSync( ) ;
        document.forms[0][0].value = VoiceObj.StgVoiceGetResultString() ;
        OnSubmit() ;
    }
</SCRIPT>
</HEAD>

<BODY onload="javascript:Listen();">
<OBJECT ID="VoiceObj" CLASSID="CLSID:5FF0CA2F-A640-4F62-8EDF-C97F30B562FB">
    <PARAM NAME="_Version" VALUE="65536">
</OBJECT>
    <P>Demo Voice with FlexTools</P>
    <FORM id="form1">
    <P>Quantity said: <INPUT type="text" id="qty" name="qty"
action="voixtreme2.htm"></P>
    <P><INPUT type="submit" onclick="javascript:OnSubmit();" value="Envier"></P>
    </FORM>
    <A HREF="voixtremel.htm">Return</A>
</BODY>
</HTML>
```

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

#### English

```
!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 ;
```