Installation Guides

The Greenlandic Martha Voice

The Martha voice has been developed by MV-Nordic for Synscenter Refsnæs and Sensus. The development has been funded by VELUX Fonden.

25-11-2014

This document describes how to install the voices on Windows, Android and Mac OS X. For iOS it describes how to implement the component.

1 Content

2 Introduction

This document describes how to install the voice on Windows, Mac OS X and Android. For iOS it describes how to implement the voice component into an app.

3 Windows installation

In the following it is assumed that the Martha installation is on a dvd.

To install the Martha voice on a single Windows pc do following:

1) Place the installation dvd into the dvd drive of the computer.
2) The installation will start automatically. If not:
    a. Open root folder of the dvd in the Windows explorer and double click om setup.exe or the msi-file.
3) The installation is made as a standard windows wizard installation. Follow the instructions of the Wizard.

The voice can be removed from the PC as any other PC program from the Control Panel in Windows.

4 Android installation

Copy the Android installation file (Android-Martha.apk) to the Android device and double click on it. The installation will start. Press 'install' when asked.

5 Mac OS X installation

Copy the Mac OS X installation file (MarthaVoice.dmg) to the Mac and double click on it. It will be opened as a pkg file. Double click on this file to install it.

6 iOS component

The iOS component has to be implemented (i.e. compiled) into an app from where the speech is initiated.

The component consists of following main parts:

- **API file**: MVSpeechSynthesizer.h:
- **Binary library file**: libMVSpeechSynthesizer.a
- **Sound data files**: mv_kl_mlj_1_ipad.data, mv_kl_mlj_2_ipad.data, scmlib.data

The API file describes the interface which must be called to use the speech synthesis. The binary (and API) file must be compiled into the app. The Sound data files is placed outside the app and accessed from the Speech Synthesis.

In order to make an iOS app you need to be certified a developer and have the appropriate development environment. Get more information from Apple (https://developer.apple.com/).

Include the API and Binary file to your iOS project and call the appropriate functions:

6.1 iOS API

The API consists of following functions and declarations:

Declarations	Usage
`@protocol MVSpeechSynthesizerDelegate;`	Forward declaration of the protocol which defines the call back functions
`@interface MVSpeechSynthesizer : NSObject <AVAudioPlayerDelegate> { AVAudioPlayer* audioPlayer; id<MVSpeechSynthesizerDelegate> delegate; }`	The singleton speech object. To be allocated by the Application
`@property (nonatomic,strong)id<MVSpeechSynthesizerDelegate> delegate;`	The delegate property which contains the callback functions
`@property (nonatomic)float rate;`	Rrate denotes the Playback rate. 1.0 provides normal playback rate. The available range is from 0.5 for half-speed playback through 2.0 for double-speed playback.

Function	Usage
`(BOOL)setVoiceWithPath:(NSString *)path;`	setVoiceWithPath sets the path to the 4 resource files needed by Martha: descriptor.txt, scmlib.data, mv_kl_mlj_1_ipad.data and mv_kl_mlj_2_ipad.data The method must be called after initialization
`(BOOL)isSpeaking;`	isSpeaking returns true if a sentence or a word currently is being spoken by the audio device
`(BOOL)startSpeakingString:(NSString *)text;`	startSpeakingString sends a sentence to the synthesizer. The method is asynchroneous. Prior to the first word being spoken, the callback function "didFinishInit" is invoked. Every time a new word is spoken, the callback function "willSpeakWord" is invoked. As parameter is conveys the character range being spoken. When the whole sentence has been spoken, the callback function “didFinishSpeaking" is invoked.
`(BOOL)SpeakString:(NSString *)text;`	SpeakString act the same way as tartSpeakingString, except that the callback function "willSpeakWord" is not used. This function is typically used when only one word is to be spoken, e.g during typing of text
`(void)pauseSpeaking;`	pauseSpeaking sets speaking on pause
`(void)continueSpeaking;`	continueSpeaking resumes speaking after pause.

Function	Usage
`(void)stopSpeaking;`	stopSpeaking aborts speaking
`(id)init;`	init initiates MVSpeechSynthesizer object. It must be called immediately after the object has been created

Callback function	Usage

`@protocol MVSpeechSynthesizerDelegate <NSObject>`
 	
`@required -`
`(void)speechSynthesizer:(MVSpeechSynthesizer *)sender didFinishSpeaking:(BOOL)finishedSpeaking;`
 	
`@required -`
`(void)speechSynthesizer:(MVSpeechSynthesizer *)sender didFinishInit:(BOOL)initiatingVoice;`
 	
`@required -`
`(void)speechSynthesizer:(MVSpeechSynthesizer *)sender willSpeakWord:(NSRange)characterRange ofString:(NSString *)string;`
 	
`@end`
	The protocol MVSpeechSynthesizerDelegate encapsulates the callback functions described above.

6.2 iOS example

Following small example shows how the component can be used.

```
// #import <UIKit/UIKit.h>
#import "MVAppDelegate.h"
#import "MVSpeechSynthesizer.h"

int main(int argc, char *argv[])
{
    @autoreleasepool {
    MVSpeechSynthesizer *obj = [MVSpeechSynthesizer alloc];

        [obj init];
        [obj setVoiceWithPath:@"/Volumes/Macintosh HD Mountin Lion/kl"];
        for (int i=0;i<10000;i++)
        {
            /*
            [obj setVoiceWithPath:@"/Volumes/Macintosh HD Mountin Lion/eng"];
            [NSThread sleepForTimeInterval:2];
            [obj SpeakString:@"How are you"];
            [NSThread sleepForTimeInterval:5];
            [obj setVoiceWithPath:@"/Volumes/Macintosh HD Mountin Lion"];
            [NSThread sleepForTimeInterval:2]; */
            char str[100];
            sprintf(str,"%d",i);
            NSString *s = [[NSString alloc ]initWithUTF8String:str];
            [obj SpeakString:s];
            [NSThread sleepForTimeInterval:3];
            [obj stopSpeaking];

        }
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([MVAppDelegate class]));
    }
    return 0;
}
```