Mobile Speech Recognizer

by Piotr Zmudzinski

ptr.zmudzinski@gmail.com

About

Wouldn't your prefer to let your users speak instead of making them type?

This plugin uses OS components for speech recognition and send it to your Unity scripts as *String* objects.

Plugin supports:
- **Android >= 3.0 (haven't tested below, it might work though…),**
- **iOS >= 10.0.**

You have to prepare fallback code to cover cases when user doesn't have access to speech recognition or when it fails (*SpeechRecognizer.EngineExists()* for the help!). Keep in mind that both iOS and Android might use Internet connection for speech detection, which means it might fail in case there's no active connection.
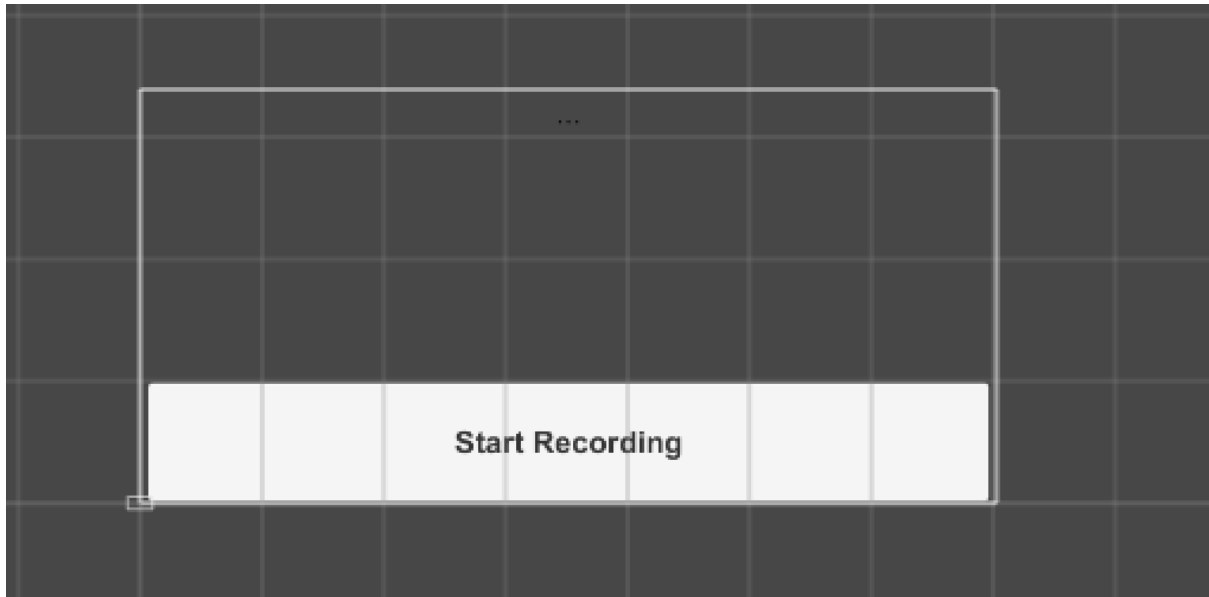
Plugin doesn't work in Editor!
You have to run your app on real iOS or Android device.

Quick Start

Open example scene

Go to *KKSpeechRecognizer/Example* folder inside Unity and open *ExampleScene*:

It shows basic usage of a plugin, which is:

1. Detecting if speech recognition exists on user's device (keep in mind that it won't be available on e.g. iOS 9 or old Android phones),
2. If it exists, and user clicks on "Start Recording" button it listens for recognized text and displays it on a screen,
3. On Android, speech recognition automatically detects when user finishes speaking, but on iOS we have to wait for user clicking "Stop Recording" to finish whole process (i.e. get final results).

Before running it on Android or iOS device you have to…

Setup permissions

iOS

After generating Xcode project (keep in mind that you have to use Xcode 8 or higher) you have to add two permissions keys to your project:

NSMicrophoneUsageDescription

explanation from Apple docs:

This key lets you describe the reason your app accesses any of the the device's microphones. When the system prompts the user to allow access, this string is displayed as part of the alert.

NSSpeechRecognitionUsageDescription

explanation from Apple docs:

This key lets you describe the reason your app sends user data to Apple's speech recognition servers. When the system prompts the user to allow access, this string is displayed as part of the alert.

You can do it in two ways:

- Automatic (default and recommended)

Open *KKSetSpeechRecognitionPermissionsoniOS* script from Editor folder inside your project. Change values of those texts:

```
public static string microphoneUsageDescription = "Put something here about microphone usage";
public static string speechRecognitionUsageDescription = "Put something here about speech recognition usage";
```

Generate your Xcode project. Your are ready.

- Manually

Disable *KKSetSpeechRecognitionPermissionsoniOS* script by either removing it or setting shouldRun variable to false.

After generating your Xcode project open *Info.plist* file and right-click on it and click *"Add row"*. Then enter two required two keys with descriptions.

In the end it should look like it:

Privacy - Microphone Usage Description		String	Your microphone will be used for speech recognition
Privacy - Speech Recognition Usage Description		String	Speech recognition will be used to detected words spoken by you

Android

You must add those two permissions to your AndroidManifest.xml:

```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.INTERNET" />
```

If you don't know how to do that check that link: http://answers.unity3d.com/questions/525838/help-about-adding-permissions-on-android.html

Now you are ready to play with the app. Press Build&Run and…

Play with speech recognition app

Click on *"Start Recording"* button and start talking. You should see live results in text above. Notice that on Android it automatically detects when you stop talking, but it rely on user when it comes to iOS.

You can start looking at *RecordingCanvas* script which covers all functionality, but if you need more details take a look at **API Overview** chapter below.

API Overview

Plugin consists of two main classes:

- *SpeechRecognizer* which lets you start/stop recording and return availability/permissions informations,

- *SpeechRecognizerListener* which sends you callback methods when speech has been detected or any error occur.

Both of them are under *KKSpeech* namespace so remember to put

```
using KKSpeech;
```

in your scripts.

Check *RecordingCanvas* script for example usage.

SpeechRecognizer

Basic usage:

```
if (SpeechRecognizer.IsRecording()) {
    SpeechRecognizer.StopIfRecording();
} else {
    SpeechRecognizer.StartRecording(true);
}
```

static bool ExistsOnDevice()

Returns *true* if speech recognition service exists on device.

!!! **You should call it before using other *SpeechRecognizer* methods and prepare your UI in case recognition system is missing. !!!**

static void RequestAccess()

This method should be used together with *onAuthorizationStatusFetched* event from *SpeechRecognizerListener*.

Behaviour differs on each platforms:

- iOS

It will show OS dialog asking user for permission to execute speech recognition (it will happen for first-time call only). Result will be returned in *onAuthorizationStatusFetched*. You should prepare yourself that user won't allow engine to run.

- Android

It isn't required to call that method on Android platform, as permissions are handled by manifest's *uses-permissions* entries. However, when you call it on Android plugin will check if manifest is set correctly and return result in *onAuthorizationStatusFetched event.*

Possible returned values (in *onAuthorizationStatusFetched* event):

```
public enum AuthorizationStatus {
        Authorized,
        Denied,
        NotDetermined,
        Restricted
}
```

- Authorized *[iOS&Android]*

on Android, it means that there's correct permission setting in manifest.
on iOS, as Apple docs say:

The user authorized your app's request to perform speech recognition.

- Denied *[iOS&Android]*

on Android it means that there's **invalid** permission setting in manifest.
on iOS, as Apple docs say:

The user denied your app's request to perform speech recognition.

- NotDetermined *[iOS only]*

from Apple docs:

The authorization status of your app's request to perform speech recognition is unknown.

- Restricted *[iOS only]*

from Apple docs:

The device denies your app's request to perform speech recognition.

*static bool **IsRecording()***
Returns *true* if speech recognition engine is currently running.

*static **AuthorizationStatus** **GetAuthorizationStatus()***
Returns current authorization status. Check *RequestAccess* description for possible returned values.

*static void **StopIfRecording()***

Stops engine from recognition process if it's currently running. Keep in mind that on Android, engine automatically stops when user stop talking so it's mostly for iOS platform use.

static void **StartRecording(*bool* shouldCollectionPartialResults)**

Starts recognition process. Pass *true* if you want to collect partial results. In order to get results use *GotPartialResult* and *GotFinalResult* from *SpeechRecognizerListener*.

SpeechRecognizerListener

In order to get callbacks from Speech Recognition you have to drag&drop *KKSpeechRecognizerListener* prefab from *KKSpeechRecognizer/Prefabs.* **Keep in mind that you cannot change name of that GameObject!**

Later on, you can register yourself to callbacks, as here:

```
SpeechRecognizerListener listener =
GameObject.FindObjectOfType<SpeechRecognizerListener>();

listener.onAuthorizationStatusFetched.AddListener(OnAuthorizationStatusFetched);
listener.onAvailabilityChanged.AddListener(OnAvailabilityChange);
listener.onErrorDuringRecording.AddListener(OnError);
listener.onErrorOnStartRecording.AddListener(OnError);
listener.onFinalResults.AddListener(OnFinalResult);
listener.onPartialResults.AddListener(OnPartialResult);
listener.onEndOfSpeech.AddListener(OnEndOfSpeech);
```

Callbacks descriptions:

onAuthorizationStatusFetched(authorizationStatus)

Check *SpeechRecognizer.RequestAccess()*

onPartialResults(string)

If you passed true into *SpeechRecognizer.startRecording* method you will receive partial recognition results as a parameter.

onFinalResults(string)

You will get final recognition results as a parameter, although behaviour differs on each platform:

- iOS

It will be called after calling *SpeechRecognizer.stopIfRecording()*, as iOS engine doesn't automatically recognizer when user stops talking.

- Android

MOBILE SPEECH RECOGNIZER - UNITY PLUGIN

It will be called after *onEndOfSpeech*, as Android automatically recognise when user stops talking.

onAvailabilityChanged(bool) *[iOS only]*
from Apple docs:

availability of the speech recognizer has changed.

It might mean that even that speech recognition engine exists on device, it became unavailable, e.g. because of network connection issues.

onErrorDuringRecording(string)
Engine failed during recording process. Explanation is passed as a parameter

onErrorOnStartRecording(string)
Engine failed before recording start. Explanation is passed as a parameter.

onEndOfSpeech() *[Android only]*
from Android docs:

Called after the user stops speaking.

You should get final results after that.

Android Error Codes

On Android, in case of error you will receive error code in message which correspond to those values (e.g. *error code 7*):

CODE	MEANING
1	Network operation timed out.
2	Other network related errors.
3	Audio recording error.
4	Server sends error status.
5	Other client side errors.
6	No speech input.
7	No recognition result matched.
8	RecognitionService busy.
9	Insufficient permissions.

Future plans

Currently, plugin uses language which is set on device. I am planning to add API to use specific language (e.g. Spanish) in recognition process. Suggestions appreciated!

Troubleshooting

Do you meet issues while using this plugin?

Do you have suggestions how to improve API?

Feel free to contact me: ptr.zmudzinski@gmail.com