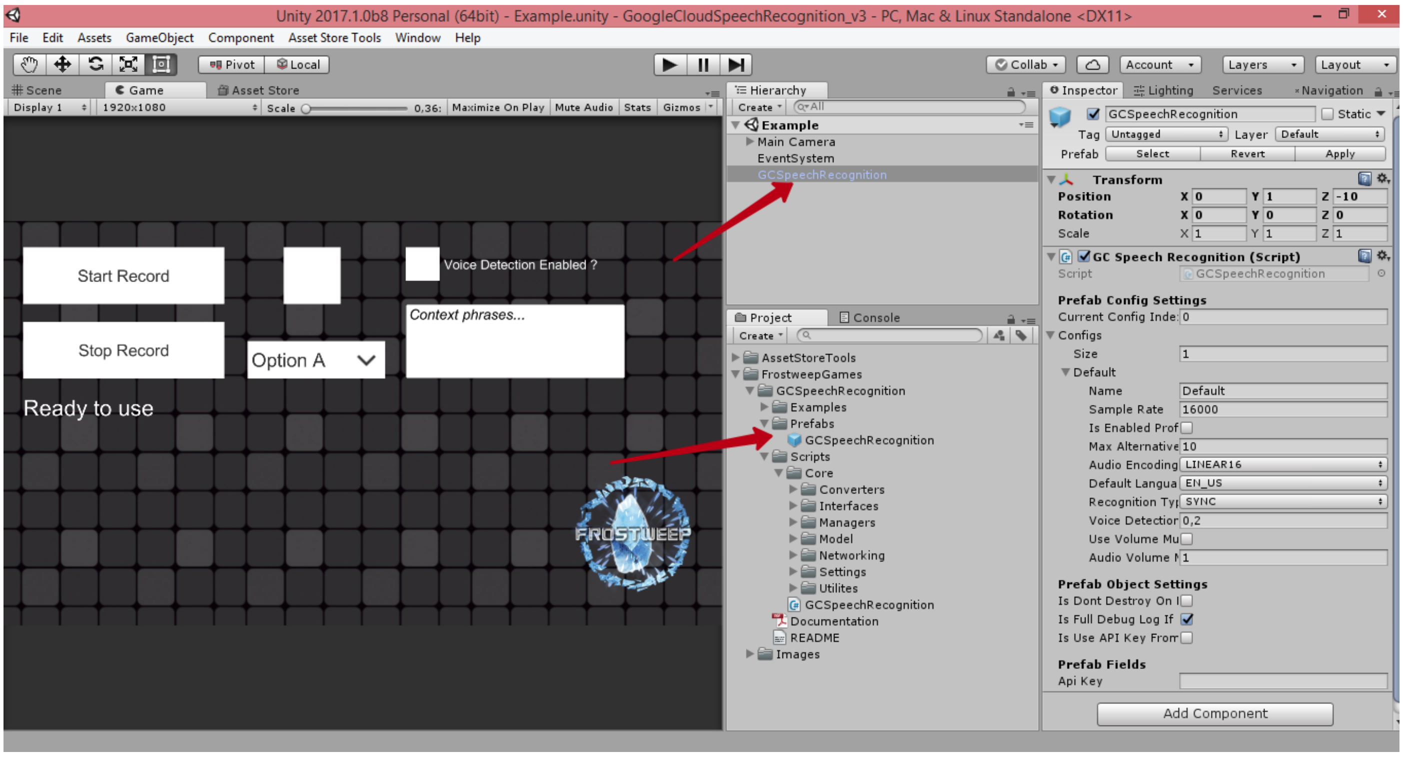
Google Cloud Speech Recognition

How to use

First of all, you need to add GCSpeechRecognition prefab from FrostweepGames->GCSpeechRecognition->Prefabs folder to your working scene.



Unity 2017.1.0b8 Personal (64bit) - Example.unity - GoogleCloudSpeechRecognition_v3 - PC, Mac & Linux Standalone <DX11>

GCSpeechRecognition ☐ Static
Tag Untagged Layer Default
Prefab Select Revert Apply

Transform
Position X 0 Y 1 Z -10
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

GC Speech Recognition (Script)
Script GCSpeechRecognition

Prefab Config Settings
Current Config Inde: 0
Configs
Size 1
Default
Name Default
Sample Rate 16000
Is Enabled Prof ☐
Max Alternative 10
Audio Encoding LINEAR16
Default Langua EN_US
Recognition Typ SYNC
Voice Detection 0,2
Use Volume Mu ☐
Audio Volume M 1

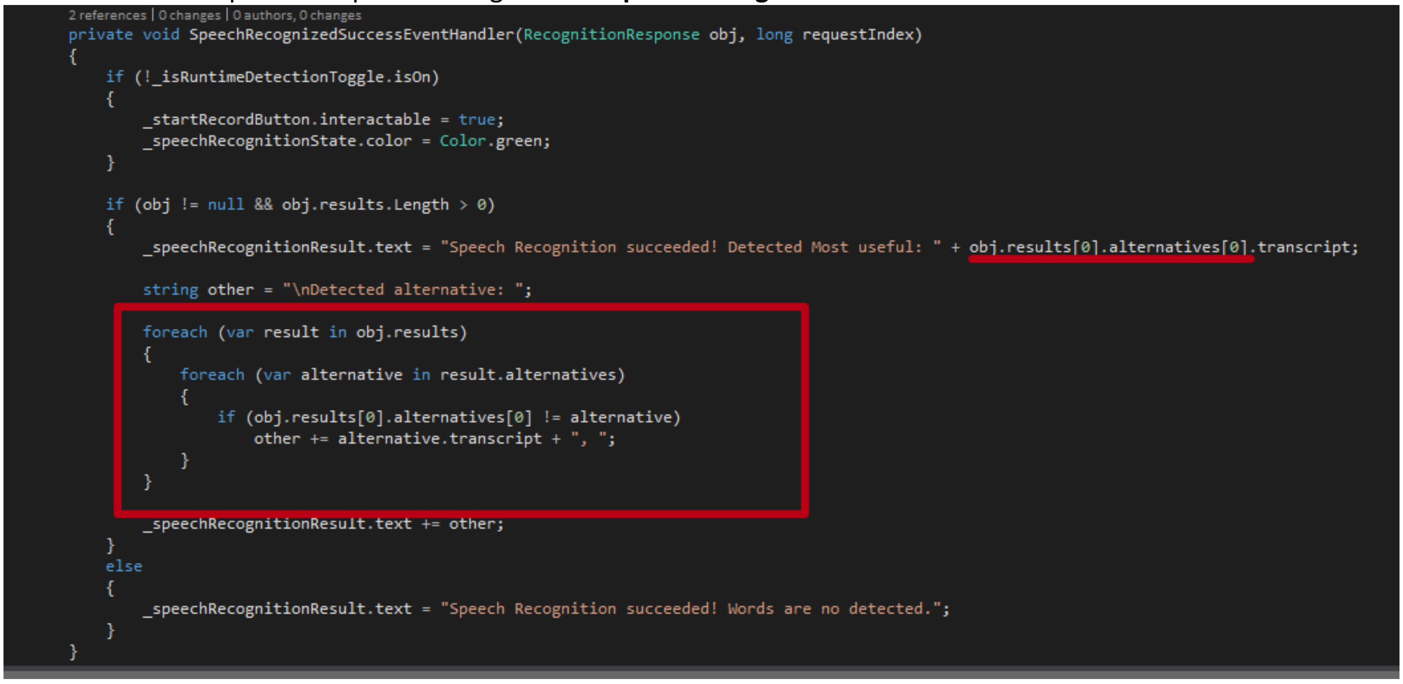
Prefab Object Settings
Is Dont Destroy On ☐
Is Full Debug Log If ✓
Is Use API Key From ☐

Prefab Fields
Api Key

Then you need to set your own API key of Google Cloud Speech Recognition into **Api Key** field and enable IsUseAPIKeyFromPrefab if you want to set api key in prefab, if not – will be used API Key from Constants.cs script.

If you don't have API Key, you can get it from https://cloud.google.com/speech/ , https://cloud.google.com/speech/docs/common/auth#restrictions

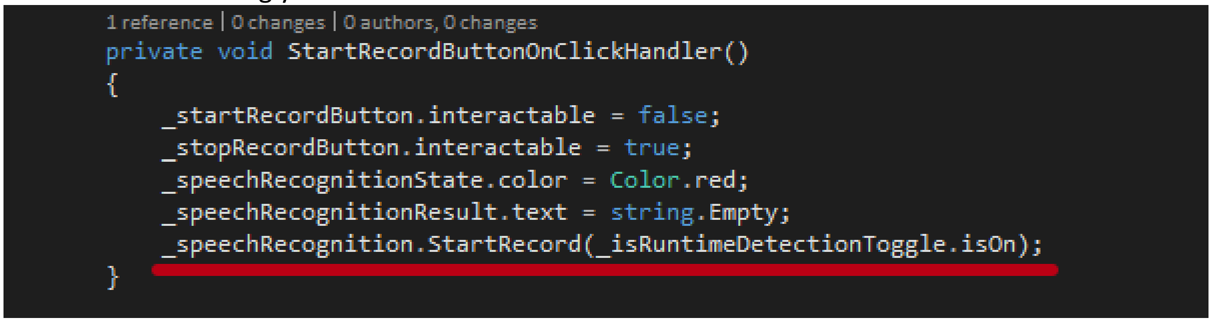
Then we need to create script with name Example and write base logic:
You can handle response of Speech Recognition in **SpeechRecognizedSuccessEventHandler**



```
2 references | 0 changes | 0 authors, 0 changes
private void SpeechRecognizedSuccessEventHandler(RecognitionResponse obj, long requestIndex)
{
    if (!_isRuntimeDetectionToggle.isOn)
    {
        _startRecordButton.interactable = true;
        _speechRecognitionState.color = Color.green;
    }

    if (obj != null && obj.results.Length > 0)
    {
        _speechRecognitionResult.text = "Speech Recognition succeeded! Detected Most useful: " + obj.results[0].alternatives[0].transcript;

        string other = "\nDetected alternative: ";

        foreach (var result in obj.results)
        {
            foreach (var alternative in result.alternatives)
            {
                if (obj.results[0].alternatives[0] != alternative)
                    other += alternative.transcript + ", ";
            }
        }
        _speechRecognitionResult.text += other;
    }
    else
    {
        _speechRecognitionResult.text = "Speech Recognition succeeded! Words are no detected.";
    }
}
```

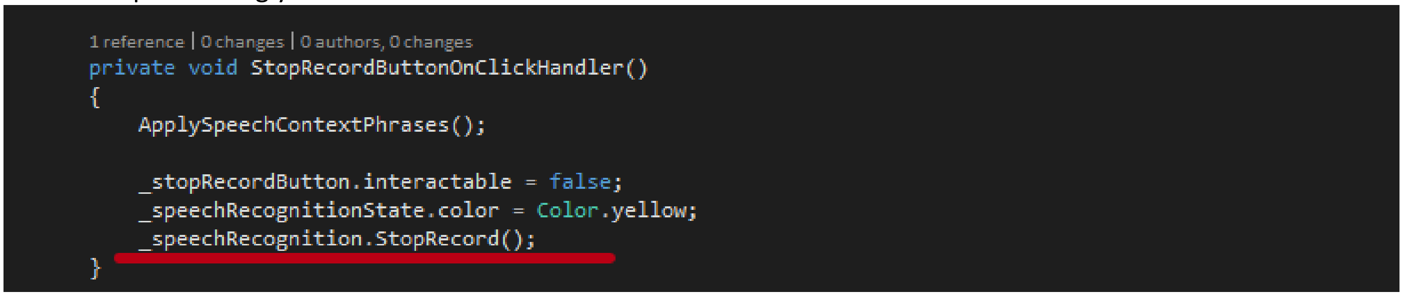
To get result of the recognition you can use RecognitionResponse->results->alternatives->transcript path. Where RecognitionResponse is an instance of the RecognitionResponse object.

For the start recording you can call this method:



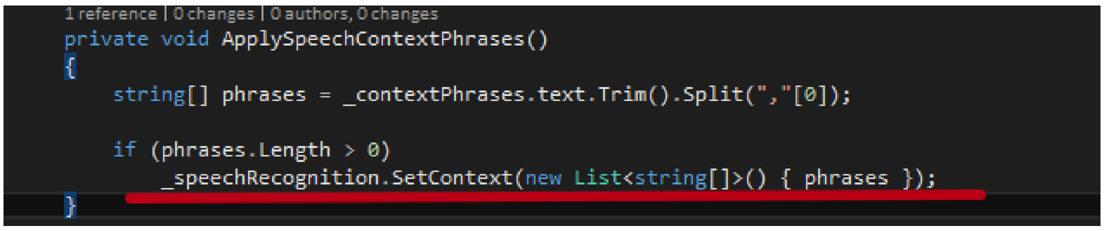
```
1 reference | 0 changes | 0 authors, 0 changes
private void StartRecordButtonOnClickHandler()
{
    _startRecordButton.interactable = false;
    _stopRecordButton.interactable = true;
    _speechRecognitionState.color = Color.red;
    _speechRecognitionResult.text = string.Empty;
    _speechRecognition.StartRecord(_isRuntimeDetectionToggle.isOn);
}
```

Include Boolean parameter for enabling runtime voice detection or not.
For the stop recording you can call this method:



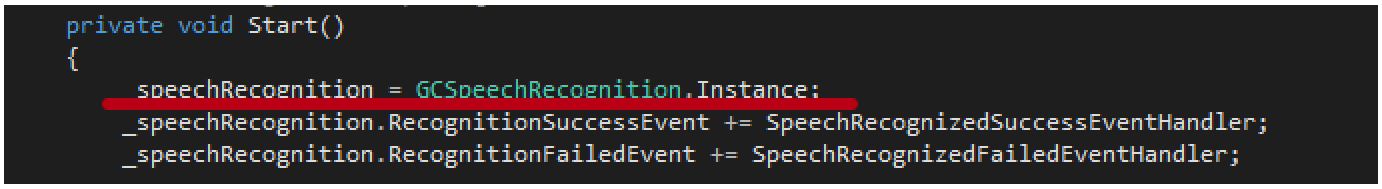
```
1 reference | 0 changes | 0 authors, 0 changes
private void StopRecordButtonOnClickHandler()
{
    ApplySpeechContextPhrases();

    _stopRecordButton.interactable = false;
    _speechRecognitionState.color = Color.yellow;
    _speechRecognition.StopRecord();
}
```

For set the list of arrays of speech contexts you can call this method:



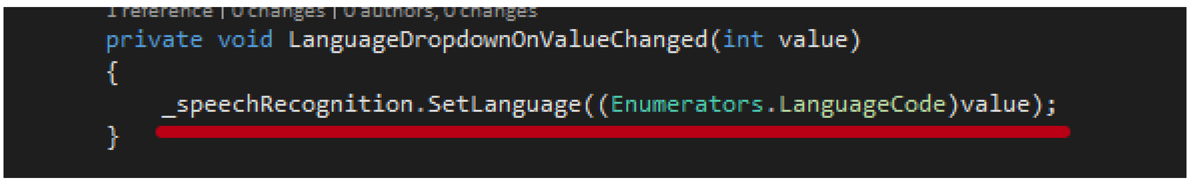
```
1 reference | 0 changes | 0 authors, 0 changes
private void ApplySpeechContextPhrases()
{
    string[] phrases = _contextPhrases.text.Trim().Split(","[0]);

    if (phrases.Length > 0)
        _speechRecognition.SetContext(new List<string[]>() { phrases });
}
```

_speechRecognition is an instance of GCSpeechRecognition class:



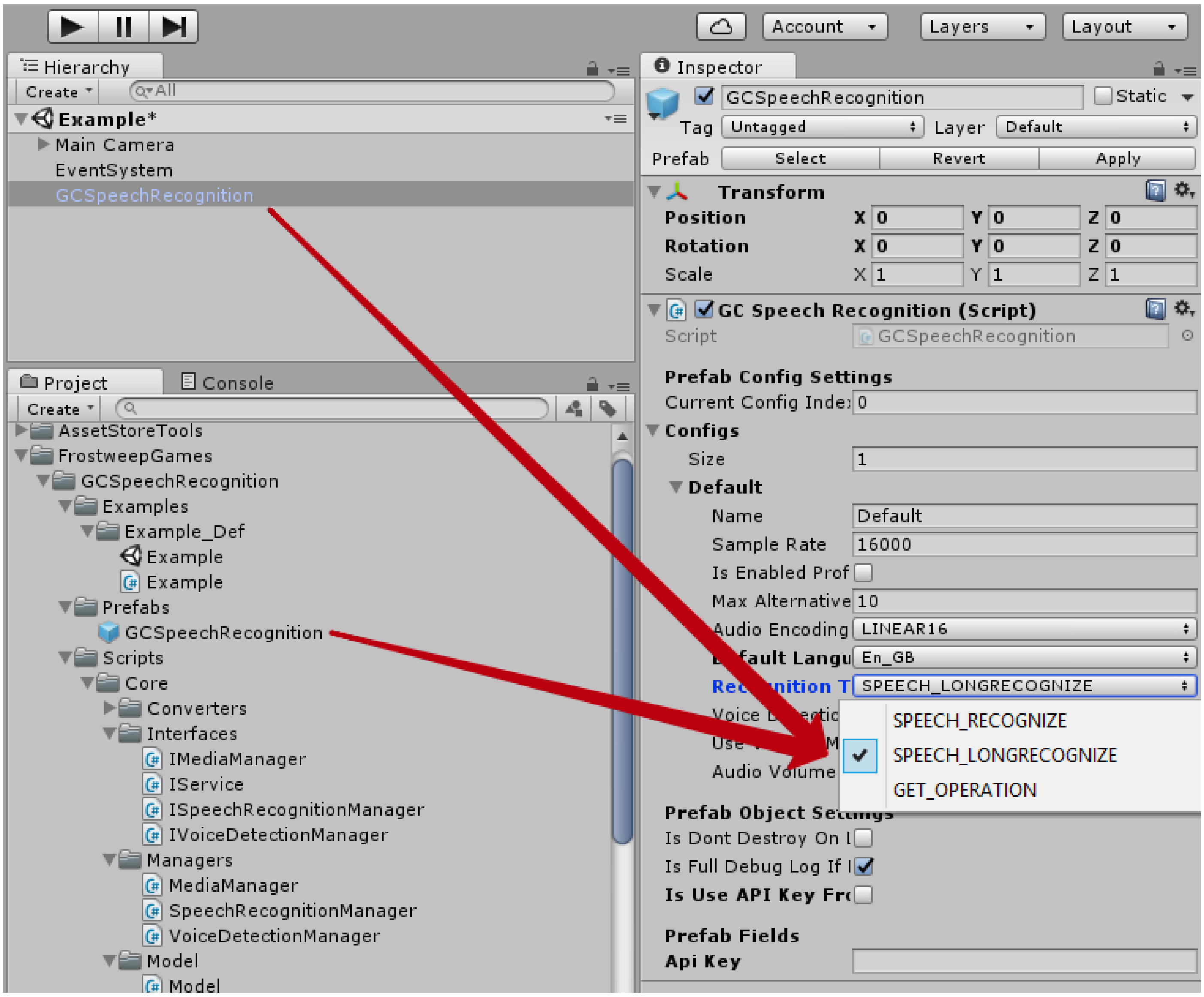
```
    private void Start()
    {
        _speechRecognition = GCSpeechRecognition.Instance;
        _speechRecognition.RecognitionSuccessEvent += SpeechRecognizedSuccessEventHandler;
        _speechRecognition.RecognitionFailedEvent += SpeechRecognizedFailedEventHandler;
```

If you want to set language you can call this method (where value is integer converted to LanguageCode enum):



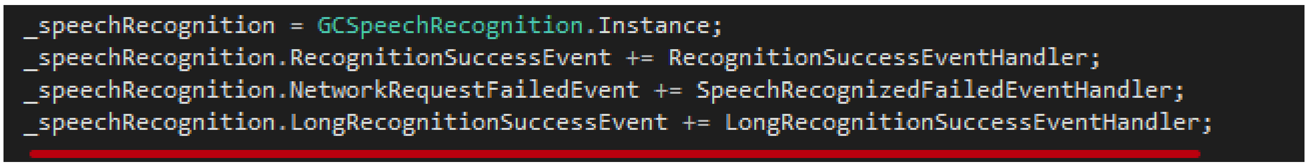
```
    1 reference | 0 changes | 0 authors, 0 changes
    private void LanguageDropdownOnValueChanged(int value)
    {
        _speechRecognition.SetLanguage((Enumerators.LanguageCode)value);
    }
```

If you want to use Long Recognize API you should choose it in config:



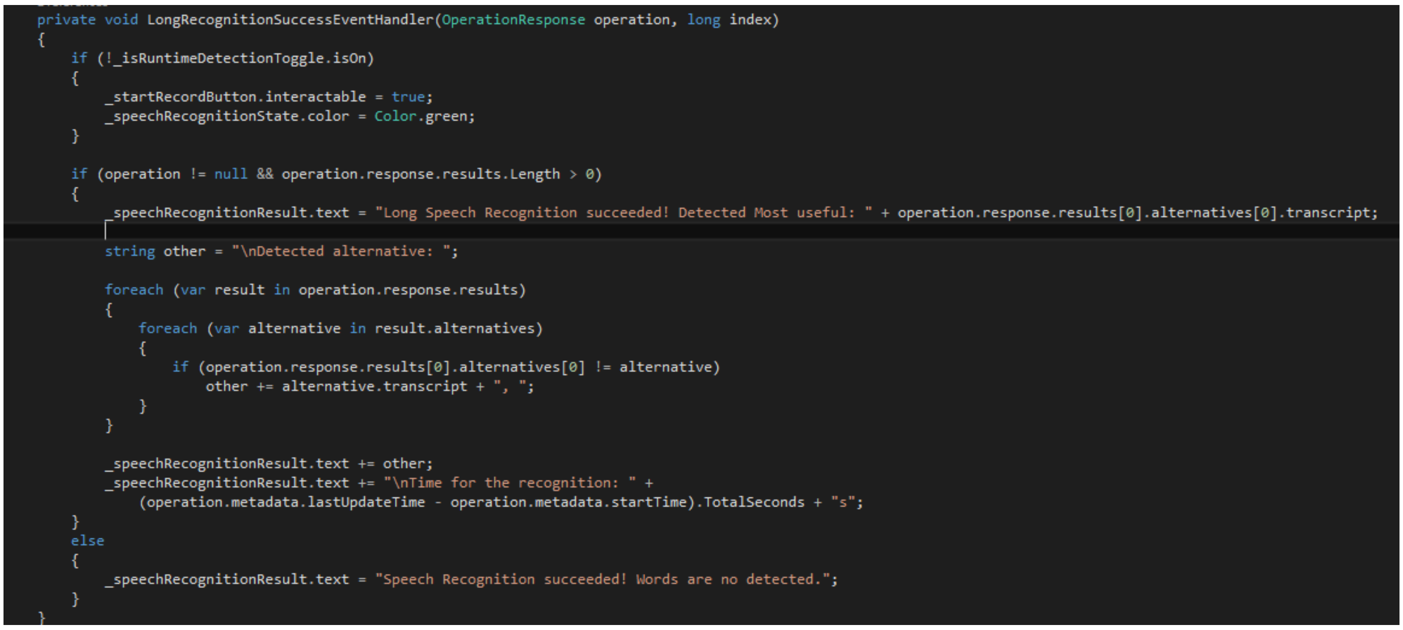
```
Hierarchy
Example*
  Main Camera
  EventSystem
  GCSpeechRecognition

Project    Console
AssetStoreTools
FrostweepGames
  GCSpeechRecognition
    Examples
      Example_Def
        Example
        Example
    Prefabs
      GCSpeechRecognition
    Scripts
      Core
        Converters
        Interfaces
          IMediaManager
          IService
          ISpeechRecognitionManager
          IVoiceDetectionManager
        Managers
          MediaManager
          SpeechRecognitionManager
          VoiceDetectionManager
        Model
          Model

Inspector
GCSpeechRecognition                          Static
Tag  Untagged        Layer  Default
Prefab  Select  Revert  Apply
Transform
Position   X 0   Y 0   Z 0
Rotation   X 0   Y 0   Z 0
Scale      X 1   Y 1   Z 1
GC Speech Recognition (Script)
Script                 GCSpeechRecognition
Prefab Config Settings
Current Config Inde: 0
Configs
  Size                 1
  Default
    Name               Default
    Sample Rate        16000
    Is Enabled Prof    [ ]
    Max Alternative    10
    Audio Encoding     LINEAR16
    Default Langu      En_GB
    Recognition T      SPEECH_LONGRECOGNIZE
    Voice Detectio
    Use
    Audio Volume
Prefab Object Settings
Is Dont Destroy On [ ]
Is Full Debug Log If [✓]
Is Use API Key Fr  [ ]
Prefab Fields
Api Key
```

For handling the Long Recognize response you should subscribe on LongRecognitionSuccessEvent:



```
_speechRecognition = GCSpeechRecognition.Instance;
_speechRecognition.RecognitionSuccessEvent += RecognitionSuccessEventHandler;
_speechRecognition.NetworkRequestFailedEvent += SpeechRecognizedFailedEventHandler;
_speechRecognition.LongRecognitionSuccessEvent += LongRecognitionSuccessEventHandler;
```

And the Handler:



```
private void LongRecognitionSuccessEventHandler(OperationResponse operation, long index)
{
    if (!_isRuntimeDetectionToggle.isOn)
    {
        _startRecordButton.interactable = true;
        _speechRecognitionState.color = Color.green;
    }

    if (operation != null && operation.response.results.Length > 0)
    {
        _speechRecognitionResult.text = "Long Speech Recognition succeeded! Detected Most useful: " + operation.response.results[0].alternatives[0].transcript;

        string other = "\nDetected alternative: ";

        foreach (var result in operation.response.results)
        {
            foreach (var alternative in result.alternatives)
            {
                if (operation.response.results[0].alternatives[0] != alternative)
                    other += alternative.transcript + ", ";
            }
        }

        _speechRecognitionResult.text += other;
        _speechRecognitionResult.text += "\nTime for the recognition: " +
            (operation.metadata.lastUpdateTime - operation.metadata.startTime).TotalSeconds + "s";
    }
    else
    {
        _speechRecognitionResult.text = "Speech Recognition succeeded! Words are no detected.";
    }
}
```

Where you can handle the results from the Operation->response->results

Example scene included to project:

FrostweepGames-> GCSpeechRecognition->Examples

Note

- Example script included in unitypackage!
- Working with il2cpp and mono
- Plugin Support Unity3D 5.3.x or above

* - Plugin doesn't support WebPlayer

Version Updates

• 3.1
- Implemented Full Long Recognize Api
- Implemented All languages that the Google Speech Recognition supporting
- Fixed bugs
- Added Newtonsoft.Json Serializer\Deserializer
- Improvements in general

• 3.0
- Updated API to the latest 1.0 version
- New code architecture (SOA)
- Added async networking (Unity Web Requests)
- Removed support of WebGL (will be added in future updates)
- Not Fully implemented Long Recognize API
- Improved Examples
- Improved Config Prefab
- Improved runtime voice detection service
- Improved media service
- Improved audio converter
- Added audio tools

• 2.1
- implemented new features
- updated and improved example
- removed 3rd party libraries

• 2.0
- UPDATED Speech Recognition API to the latest Google Cloud Speech API
- implemented new features
- implemented speech detection threshold
- changed namespaces

- fixed bugs

• 1.1
- Changed Code Namespace with FrostweepGames.SpeechRecognition on FrostweepGames.SpeechRecognition.Google
- Implemented Runtime Speech Detection Utility