Google Cloud Text To speech

- Intro:

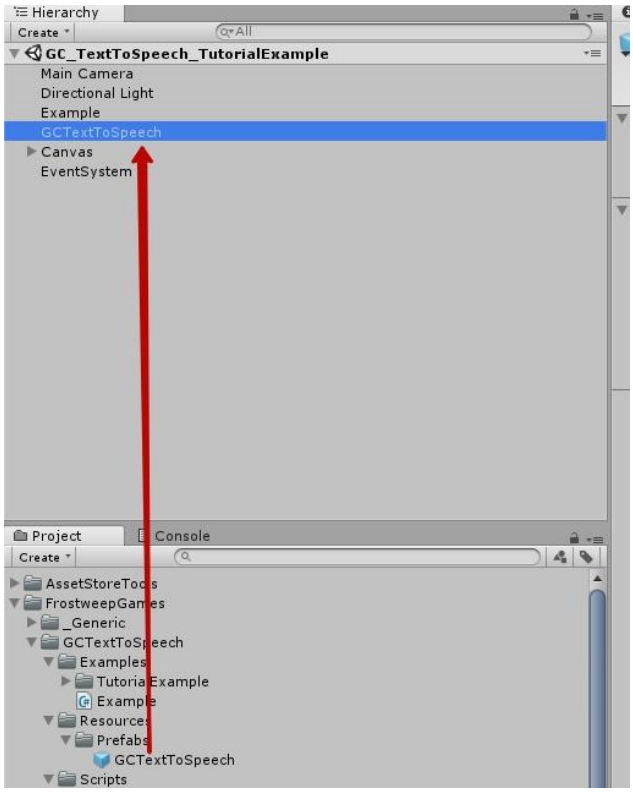
Google Cloud Text To Speech API enables developers to synthesize natural-sounding speech with 30 voices, available in multiple languages and variants. It applies DeepMind's groundbreaking research in WaveNet and Google's powerful neural networks to deliver the highest fidelity possible. With this easy-to-use API, you can create lifelike interactions with your users, across many applications and devices. How to use:

Create you first an app example:

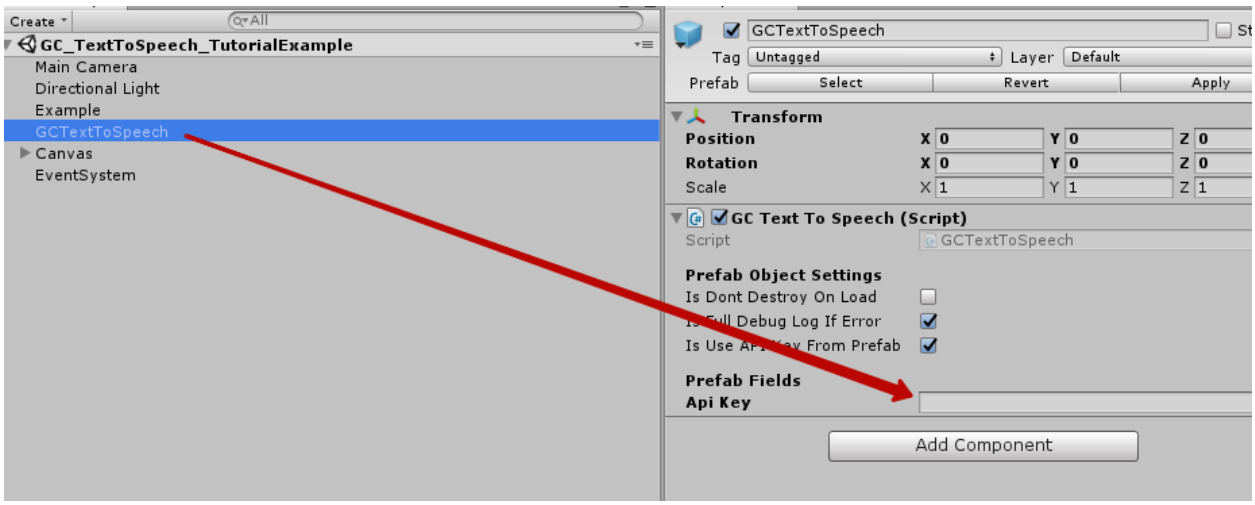
Create the script with and name it 'TutorialExample':

```
using System.Collections.Generic;
using UnityEngine;

public class Example : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

    }
}
```

Drag n Drop the Prefab of GCTextToSpeech into the scene:



Insert your own Google Cloud API Key into this field:



Create the variable for the GCTextToSpeech and get an instance of an object:

```
public class GC_TextToSpeech_TutorialExample : MonoBehaviour
{
    private GCTextToSpeech _gcTextToSpeech;


    private void Start()
    {
        _gcTextToSpeech = GCTextToSpeech.Instance;
```

Then you should subscribe on the events:

```
    _gcTextToSpeech.GetVoicesSuccessEvent += _gcTextToSpeech_GetVoicesSuccessEvent;
    _gcTextToSpeech.SynthesizeSuccessEvent += _gcTextToSpeech_SynthesizeSuccessEvent;

    _gcTextToSpeech.GetVoicesFailedEvent += _gcTextToSpeech_GetVoicesFailedEvent;
    _gcTextToSpeech.SynthesizeFailedEvent += _gcTextToSpeech_SynthesizeFailedEvent;
```

```
    private void _gcTextToSpeech_SynthesizeFailedEvent(string error)
    {
        Debug.Log(error);
    }

    private void _gcTextToSpeech_GetVoicesFailedEvent(string error)
    {
        Debug.Log(error);
    }

    #endregion failed handlers

    #region sucess handlers

    private void _gcTextToSpeech_SynthesizeSuccessEvent(PostSynthesizeResponse response)
    {
        audioSource.clip = _gcTextToSpeech.GetAudioClipFromBase64(response.audioContent, Constants.DEFAULT_AUDIO_ENCODING);
        audioSource.Play();
    }

    private void _gcTextToSpeech_GetVoicesSuccessEvent(GetVoicesResponse response)
    {
        _voices = response.voices;

        FillVoicesList();
    }
```

Create the GetVoices request:

```
_gcTextToSpeech.GetVoices(new GetVoicesRequest()
{
    languageCode =  _gcTextToSpeech.PrepareLanguage((Enumerators.LanguageCode)languageCodesDropdown.value)
});
```

Where **LanguageCode** is the languageCode of the voice.

When the GetVoices request will be successful, will be fire the `GetVoicesSuccessEvent`.

Handle the Voices and fill voices list into dropdown:

```
private void _gcTextToSpeech_GetVoicesSuccessEvent(GetVoicesResponse response)
{
    _voices = response.voices;

    FillVoicesList();
}
```

```
private void FillVoicesList()
{
    if (_voices == null)
        return;

    List<string> elements = new List<string>();

    for (int i = 0; i < _voices.Length; i++)
    {
        if (_voices[i].name.ToLower().Contains(((Enumerators.VoiceType)voiceTypesDropdown.value).ToString().ToLower()))
            elements.Add(_voices[i].name);
    }

    voicesDropdown.ClearOptions();
    voicesDropdown.AddOptions(elements);

    if (elements.Count > 0)
    {
        voicesDropdown.value = 0;
        VoiceSelectedDropdownOnChangedHandler(0);
    }
}
```

```
    private Voice[] _voices;
    private Voice _currentVoice;

    public Button synthesizeButton;
    public Button getVoicesButton;

    public InputField contentInputFioeld;
    public InputField pitchInputField;
    public InputField speakingRateInputField;

    public Toggle ssmlToggle;

    public Dropdown languageCodesDropdown;
    public Dropdown voiceTypesDropdown;
    public Dropdown voicesDropdown;
```

To fill the voice types into dropdown you should do that:

```
length = Enum.GetNames(typeof(Enumerators.VoiceType)).Length;
elements.Clear();

for (int i = 0; i < length; i++)
    elements.Add(((Enumerators.VoiceType)i).ToString());

voiceTypesDropdown.ClearOptions();
voiceTypesDropdown.AddOptions(elements);
voiceTypesDropdown.value = 0;
```

When the GetVoices request will be failed, will be fire the `GetVoicesFailedEvent`.

Handle the event:

```
private void _gcTextToSpeech_GetVoicesFailedEvent(string error)
{
    Debug.Log(error);
}
```

For Synthesize text you have to create request method.. lets add new button and subscribe on OnClick event:

```
public Button synthesizeButton;


private void Start()
{
    _gcTextToSpeech = GCTextToSpeech.Instance;

    _gcTextToSpeech.GetVoicesSuccessEvent += _gcTextToSpeech_GetVoicesSuccessEvent;
    _gcTextToSpeech.SynthesizeSuccessEvent += _gcTextToSpeech_SynthesizeSuccessEvent;

    _gcTextToSpeech.GetVoicesFailedEvent += _gcTextToSpeech_GetVoicesFailedEvent;
    _gcTextToSpeech.SynthesizeFailedEvent += _gcTextToSpeech_SynthesizeFailedEvent;

    synthesizeButton.onClick.AddListener(SynthesizeButtonOnClickHandler);
```

Our Handler is:

```
string content = contentInputFioeld.text;

if (string.IsNullOrEmpty(content) || _currentVoice == null)
    return;

_gcTextToSpeech.Synthesize(content, new VoiceConfig()
{
    gender = _currentVoice.ssmlGender,
    languageCode = _currentVoice.languageCodes[0],
    name = _currentVoice.name
},
ssmlToggle.isOn,
double.Parse(pitchInputField.text),
double.Parse(speakingRateInputField.text),
_currentVoice.naturalSampleRateHertz);
```

You can insert parameters:

gender – Gender of the voice from voices list

languageCode – language of the voice

name – voice name

ssml – parameter of the text format is ssml (about ssml (https://cloud.google.com/text-to-speech/docs/ssml)

pitch – pitch rate, by default 1.0

speakingRate – speaking rate, by default 1.0

sampleRateHertz – sample rate of the voice, by default uses selected voice sample rate or 16000(in Constants)

In the handler of the Synthesize request you have write this code:

```
private void _gcTextToSpeech_SynthesizeSuccessEvent(PostSynthesizeResponse response)
{
    audioSource.clip = _gcTextToSpeech.GetAudioClipFromBase64(response.audioContent, Constants.DEFAULT_AUDIO_ENCODING);
    audioSource.Play();
}
```

We are giving the content of the audio to converter with special encoding (for us it's a Linear16) and gets the AudioClip that we inserting into AuioSource and call play method.

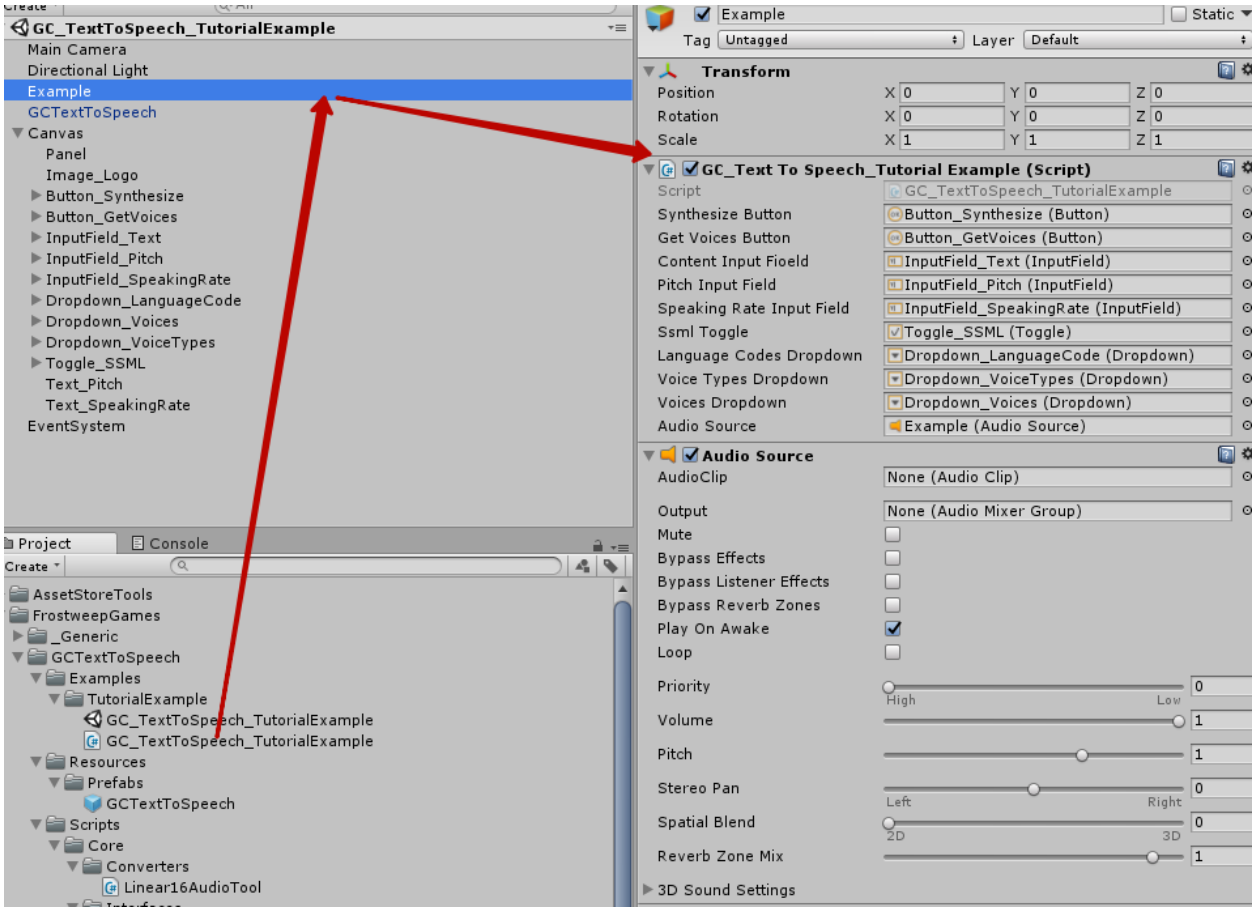
If we got failed request we can handle it like this:

```
_gcTextToSpeech.SynthesizeFailedEvent += _gcTextToSpeech_SynthesizeFailedEvent;
```

```
private void _gcTextToSpeech_GetVoicesFailedEvent(string error)
{
    Debug.Log(error);
}
```

Then:

Create an object in the scene and attach the TutorialExample script into this object:



Then connect all buttons, input fields and dropdown's from the scene into script via dragNdrop.

Also attach AudioSource component on this object to play the synthesized audio..

Then enjoy your project!

Special Parameters in constants:

```
        public const Enumerators.AudioEncoding DEFAULT_AUDIO_ENCODING = Enumerators.AudioEncoding.LINEAR16;
        public const double DEFAULT_SAMPLE_RATE = 16000;
        public const double DEFAULT_VOLUME_GAIN_DB = 0.0;
    }
```

- Note:

1) The plugin does not cover the cost of the Google Cloud Service
2) Be sure to read the terms of service of Google Cloud Text To Speech API
3) Our Asset supporting only LINEAR16 encoding of the audio. You can implement own codecs by own hand.

- Versions changes:

1.0 – Implemented Google Cloud Text To Speech API