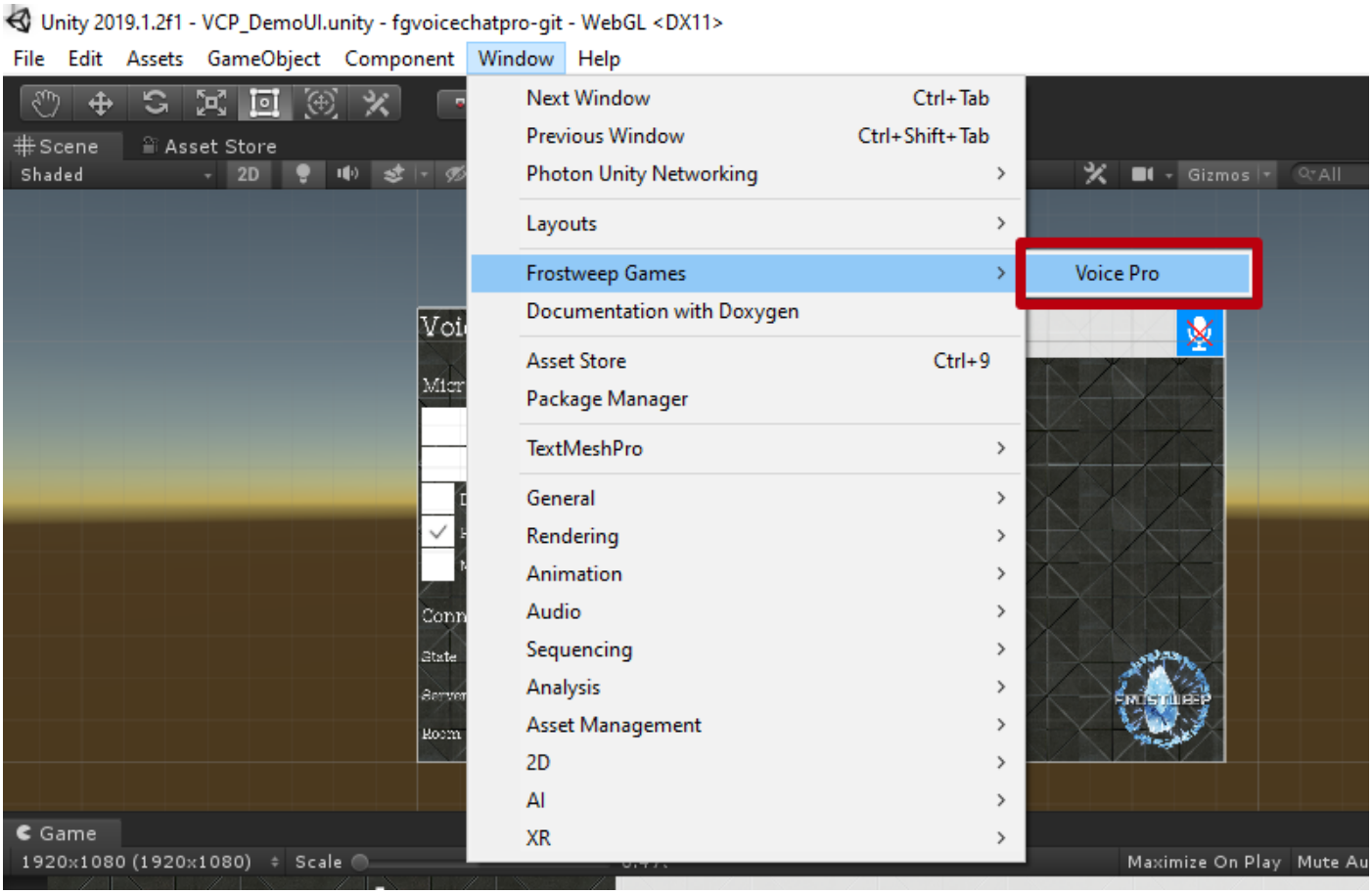
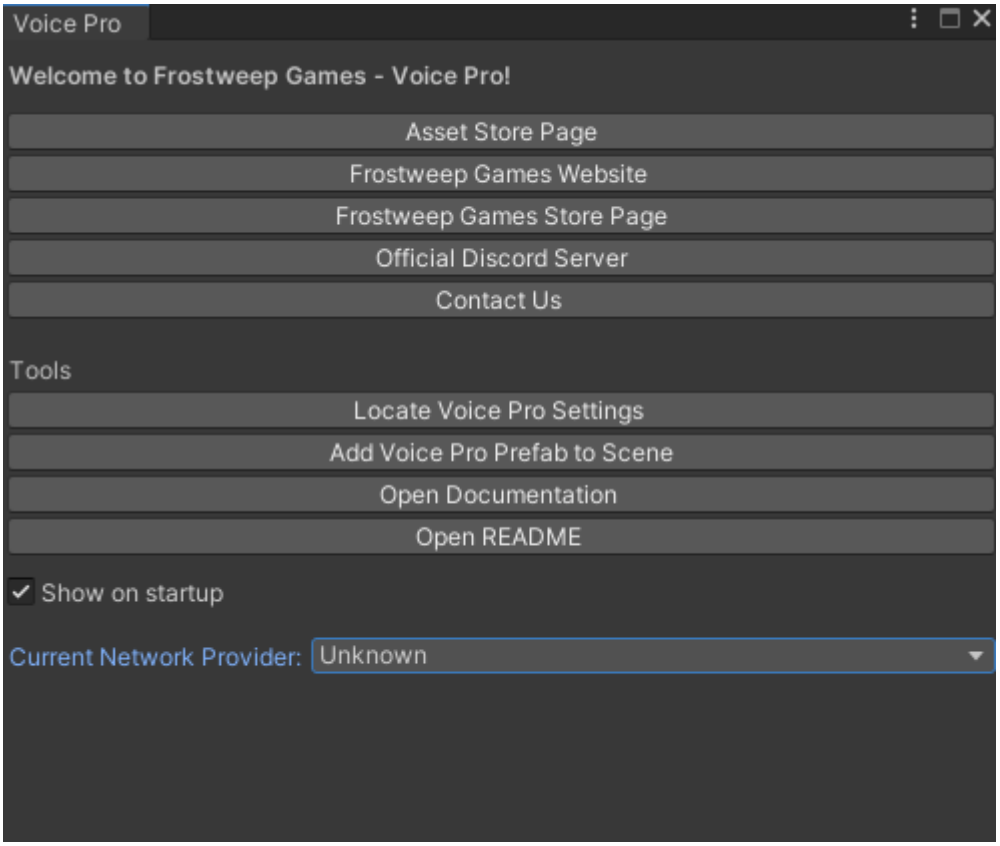
Voice Pro

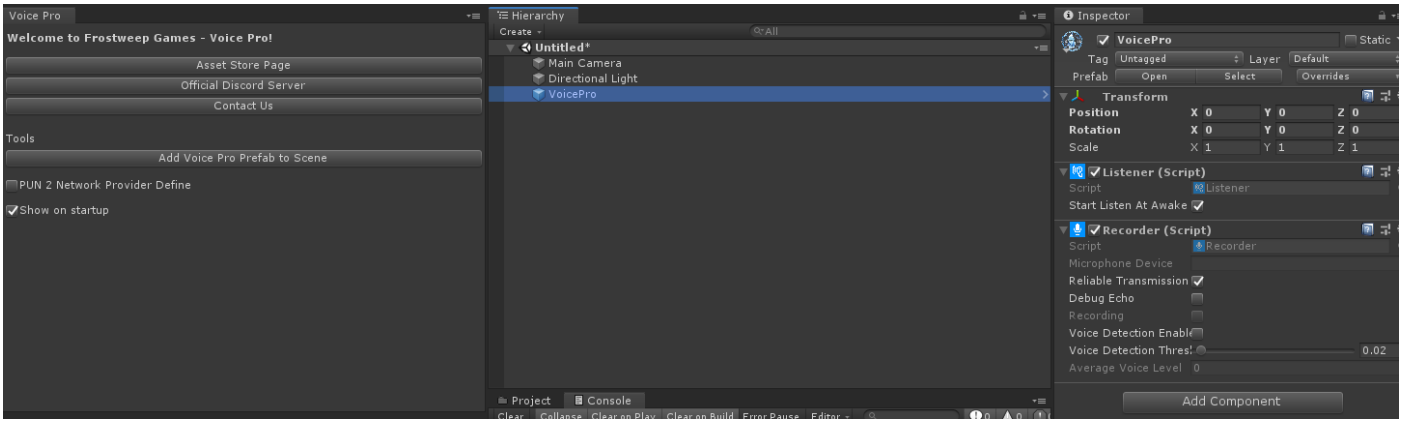
How to use

First of all open Voice Pro Welcome window if it closed:



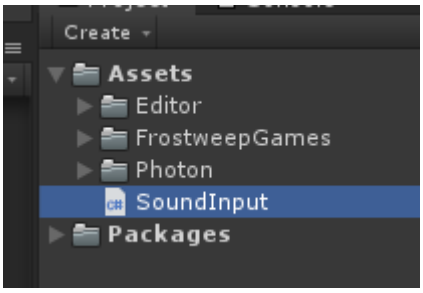


Create a new scene and click on *Add Voice Pro Prefab to Scene* button, it will create a new VoicePro object in scene.



Now you ready to implement simple functionality for sound input.

Create a new script with name SoundInput:



Then open it in scripts editor:

SoundInput.cs

Assembly-CSharp | SoundInput | Start()

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
    Скрипт Unity | Ссылок: 0
5   public class SoundInput : MonoBehaviour
6   {
7       // Start is called before the first frame update
        Сообщение Unity | Ссылок: 0
8       void Start()
9       {
10
11      }
12
13      // Update is called once per frame
        Сообщение Unity | Ссылок: 0
14      void Update()
15      {
16
17      }
18  }
19
```

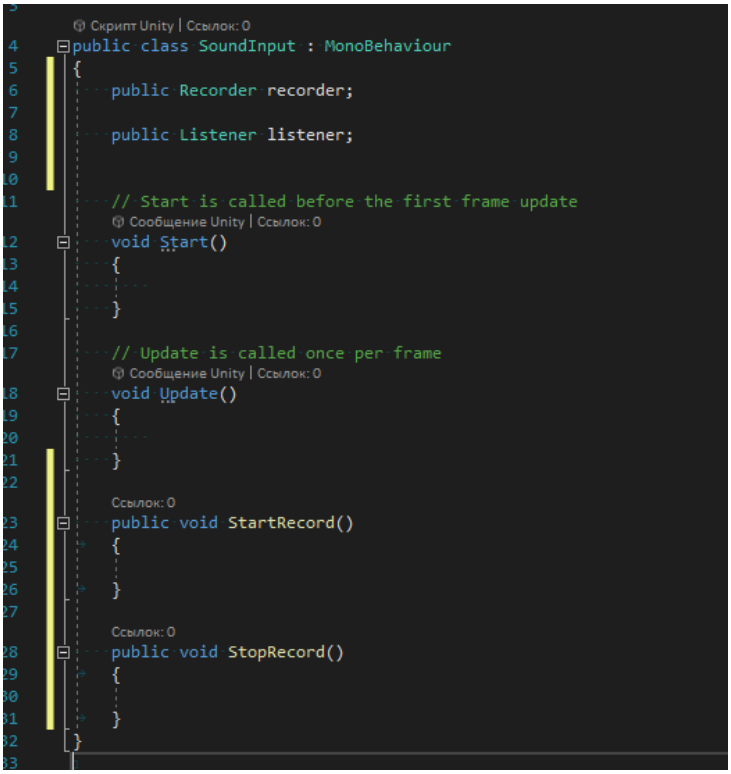
Now we have to declare two variables for recorder and listener:

Assembly-CSharp | SoundInpu

```
1   using FrostweepGames.VoicePro;
2   using UnityEngine;
3
    Скрипт Unity | Ссылок: 0
4   public class SoundInput : MonoBehaviour
5   {
6       public Recorder recorder;
7
8       public Listener listener;
9
10
11      // Start is called before the first frame update
        Сообщение Unity | Ссылок: 0
12      void Start()
13      {
14
15      }
16
17      // Update is called once per frame
        Сообщение Unity | Ссылок: 0
18      void Update()
19      {
20
21      }
22  }
23
```

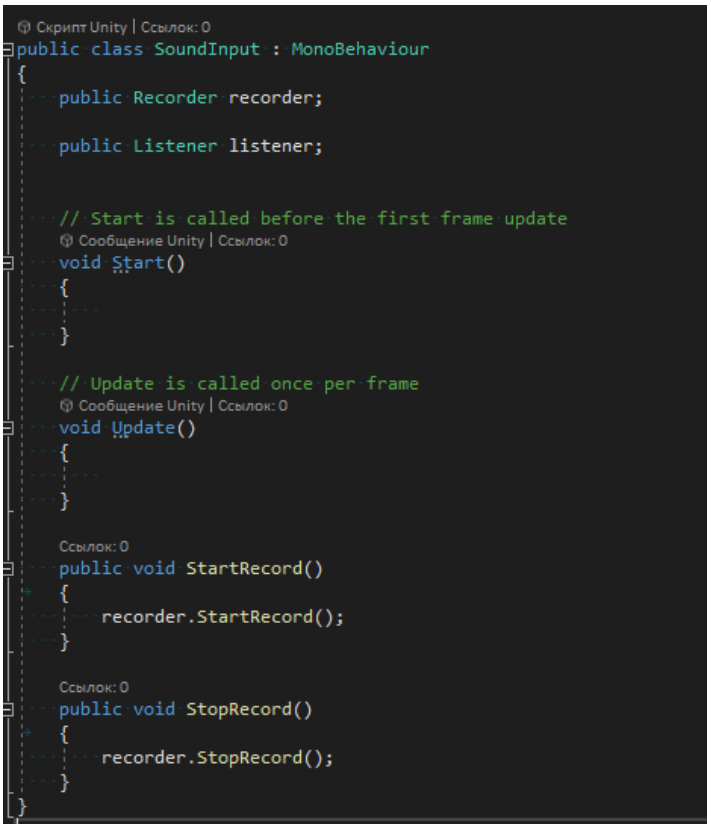
Recorder class need for recording sound from microphone. **Listener** for receiving list of available speakers (Look at VCP_Demo how it could be used).

Now need to write two functions for start record and stop it. Lets name them accordingly.



```
Скрипт Unity | Ссылок: 0
public class SoundInput : MonoBehaviour
{
    public Recorder recorder;

    public Listener listener;


    // Start is called before the first frame update
    Сообщение Unity | Ссылок: 0
    void Start()
    {

    }

    // Update is called once per frame
    Сообщение Unity | Ссылок: 0
    void Update()
    {

    }

    Ссылок: 0
    public void StartRecord()
    {

    }

    Ссылок: 0
    public void StopRecord()
    {

    }
}
```

In StartRecord function we have to call record function from Recorder. And in StopRecord call stop record function from Recorder.



```
Скрипт Unity | Ссылок: 0
public class SoundInput : MonoBehaviour
{
    public Recorder recorder;

    public Listener listener;


    // Start is called before the first frame update
    Сообщение Unity | Ссылок: 0
    void Start()
    {

    }

    // Update is called once per frame
    Сообщение Unity | Ссылок: 0
    void Update()
    {

    }

    Ссылок: 0
    public void StartRecord()
    {
        recorder.StartRecord();
    }

    Ссылок: 0
    public void StopRecord()
    {
        recorder.StopRecord();
    }
}
```

Now we should add a new variable named as isRecording with Boolean type to understand do we record now or not:

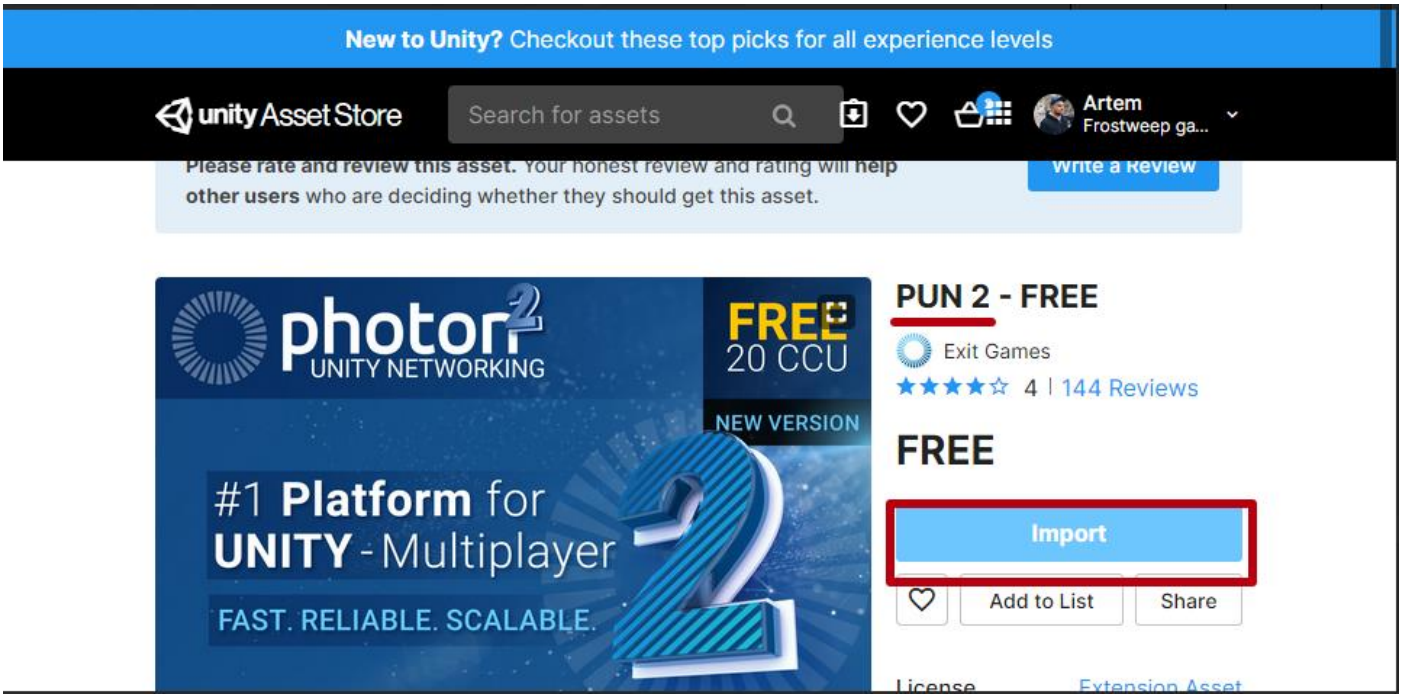
```
Скрипт Unity | Ссылок: 0
4   public class SoundInput : MonoBehaviour
5   {
6       public Recorder recorder;
7
8       public Listener listener;
9
10      public bool isRecording;
11
12
```

Lets add a logic that will start and stop record dependently from pressing of a button with use of isRecording variable:

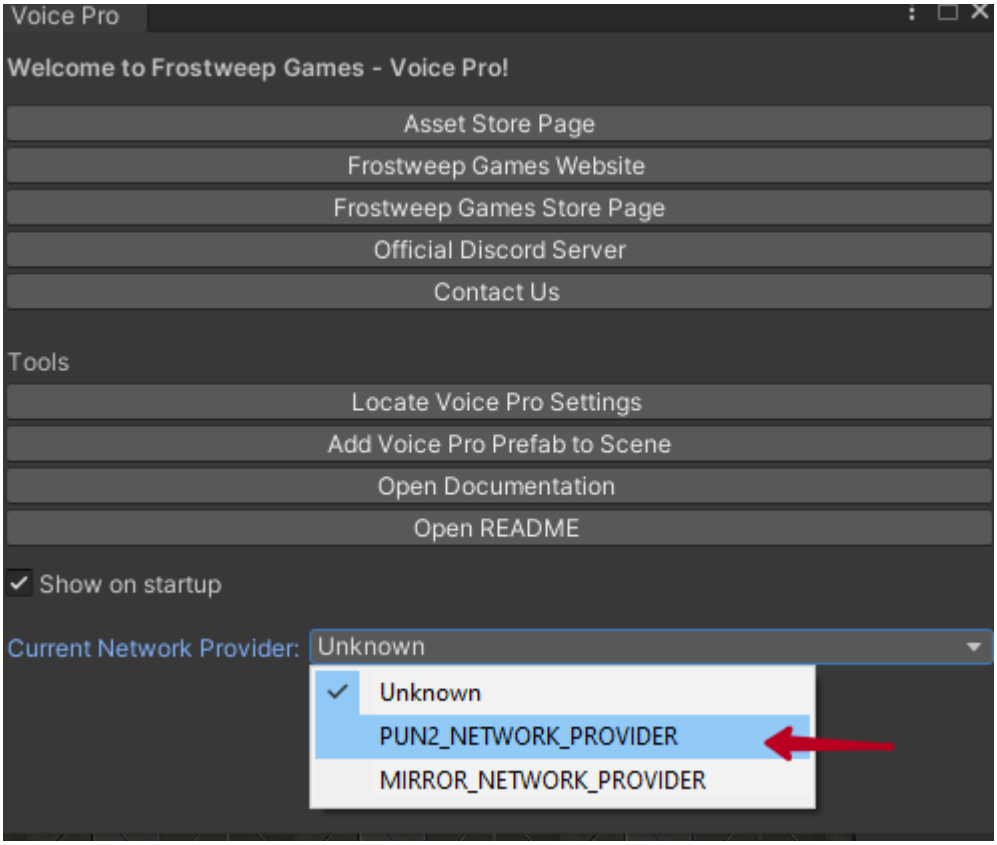
```
namespace FrostweepGames.VoicePro.Examples
{
    public class SoundInput : MonoBehaviour
    {
        public Recorder recorder;

        public Listener listener;

        public bool isRecording;

        void Update()
        {
            if (Input.GetKeyDown(KeyCode.R) && !isRecording)
            {
                StartRecord();
            }
            else if (Input.GetKeyUp(KeyCode.R) && isRecording)
            {
                StopRecord();
            }
        }

        public void StartRecord()
        {
            if (CustomMicrophone.HasConnectedMicrophoneDevices())
            {
                recorder.SetMicrophone(CustomMicrophone.devices[0]);
                isRecording = recorder.StartRecord();

                Debug.Log("Record started: " + isRecording);
            }
            else
            {
                recorder.RefreshMicrophones();
            }
        }


        public void StopRecord()
        {
            isRecording = false;
            recorder.StopRecord();

            Debug.Log("Record ended");
        }
    }
}
```

As it's a network solution you have to register networking. Currently our solution provides api for usage of PUN 2 network (**no** PUN Voice).

Lets import it from an asset store:

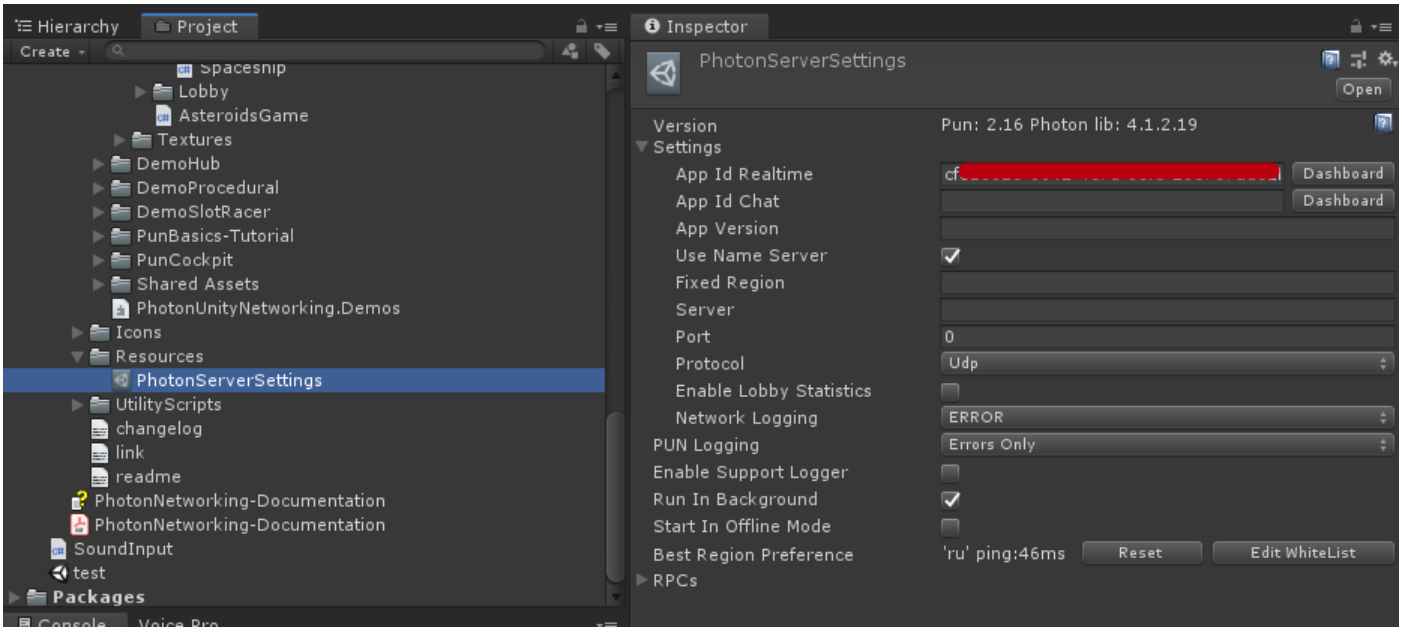


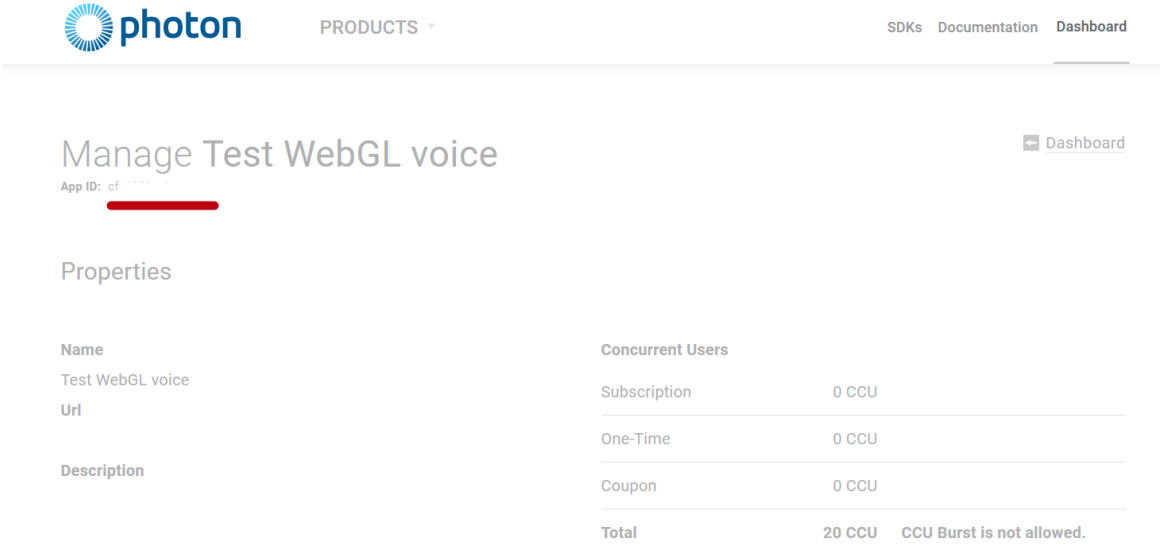
When imported you have to open Voice Pro Welcome window (described at begin of document) and select PUN 2 Network Provider:



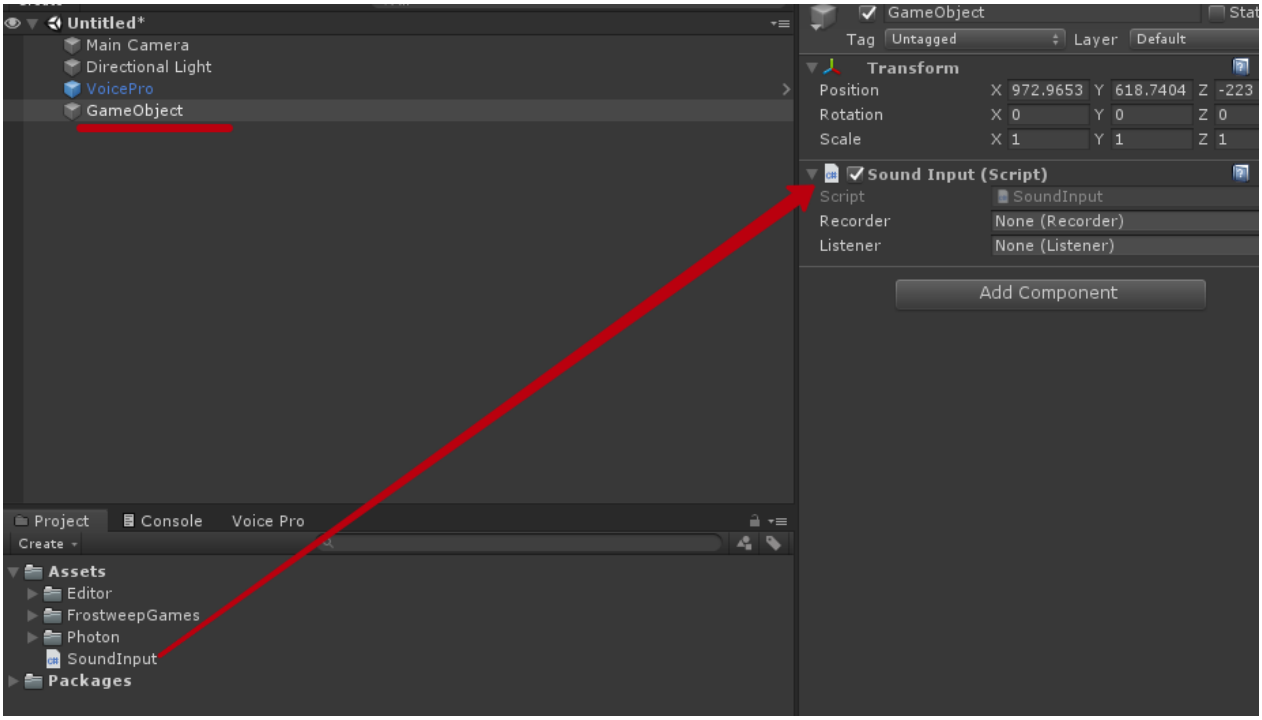
It will add special code define to list of Defines in project settings.

Don’t forget to setup PhotonServerSettings. (You have to create Photon account and register an app)

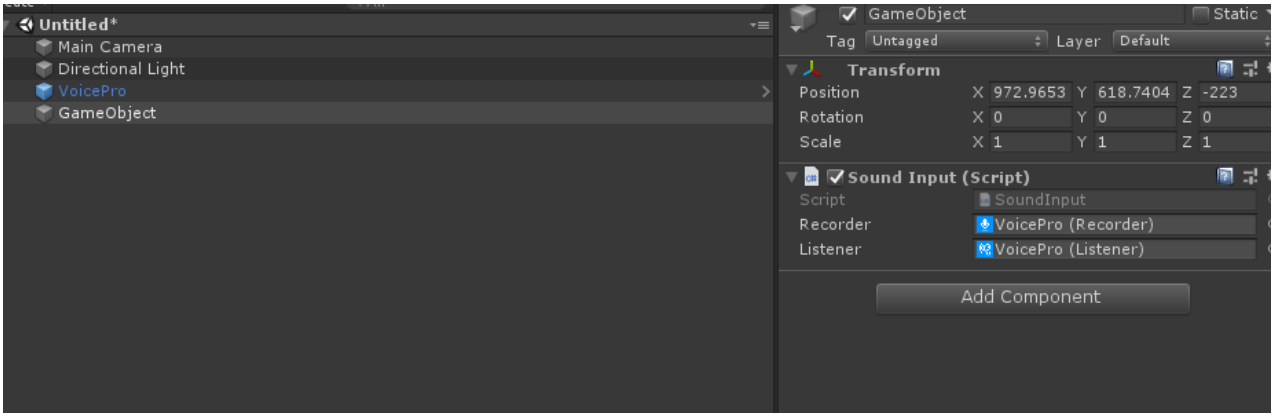




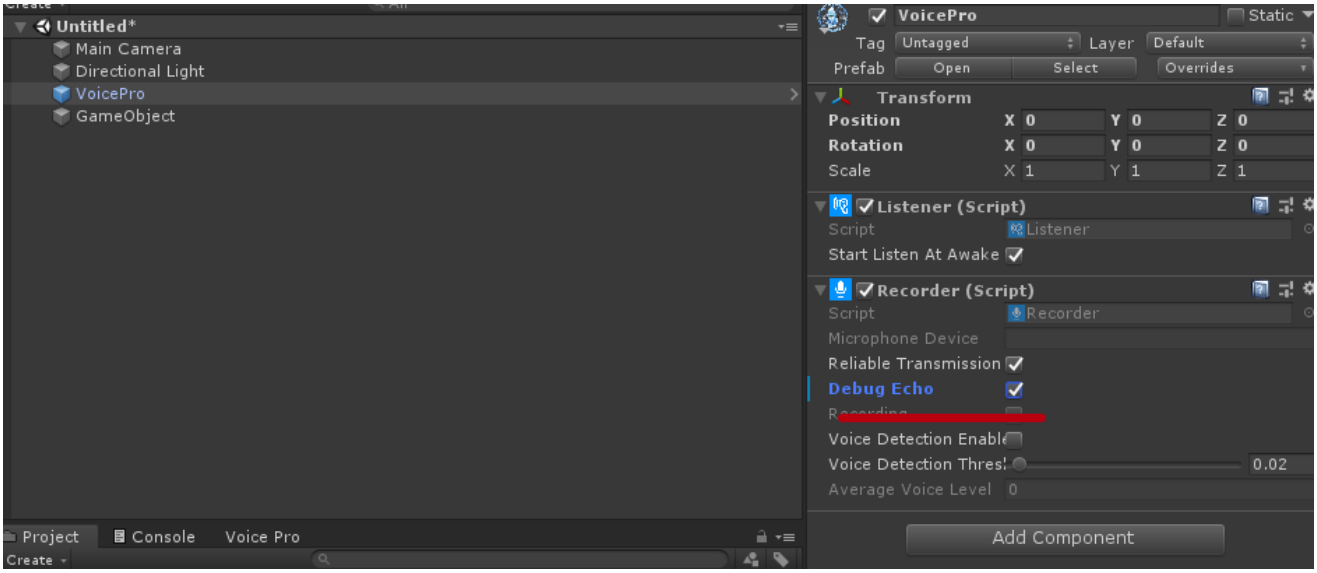
Last things that we have to do its create an empty object on created scene and attach SoundInput script we created on it(don't forget to save script before).



Now we have to set our variables. Lets drag & drop VoicePro object from scene to inspector on both fields:

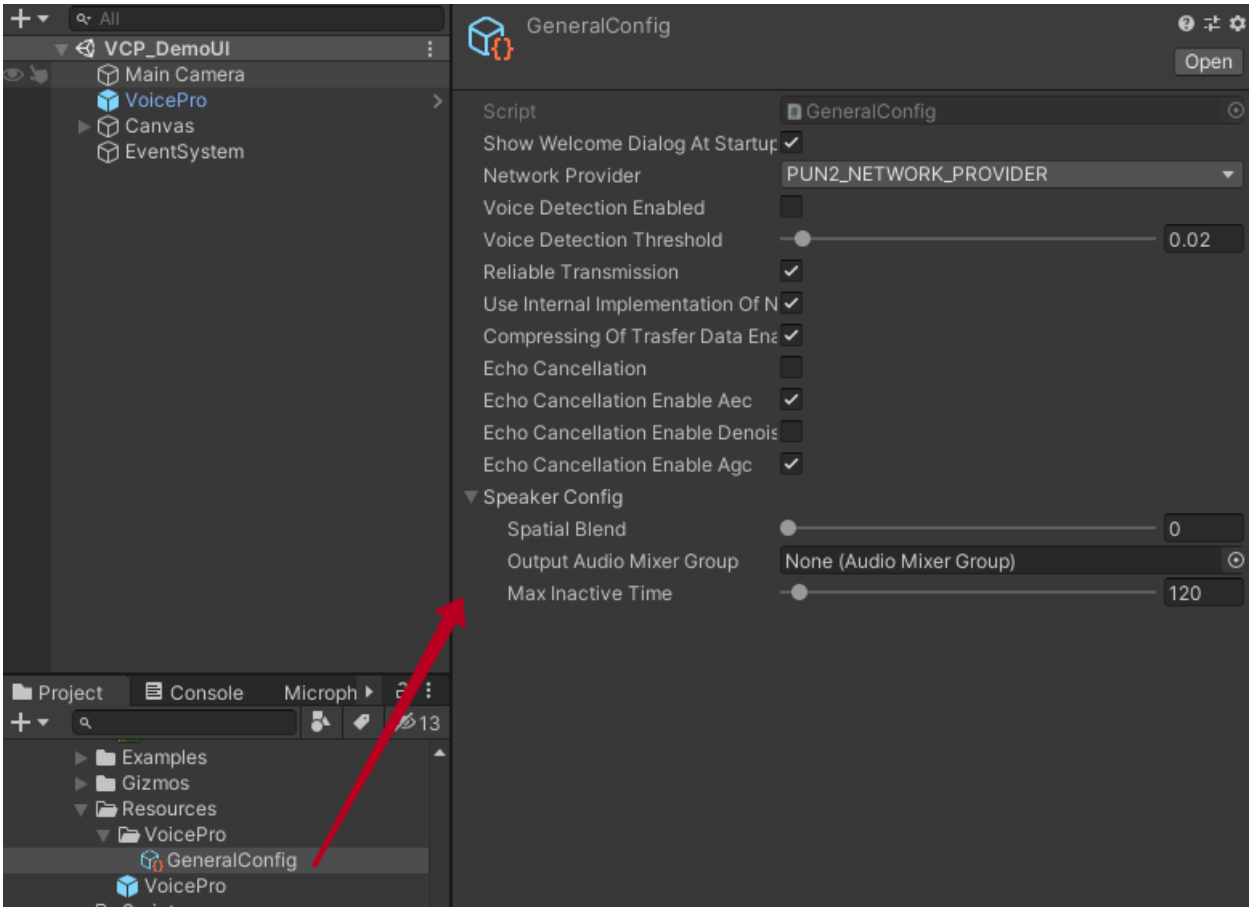


To make local test you could enable DebugEcho on Recorder component. Lets check DebugEcho checkbox:



Now save scene in project folder and play scene. During pressing on R button, - it will record and transmit data.

Also you could configure settings of Voice Pro by editing GeneralConfig:



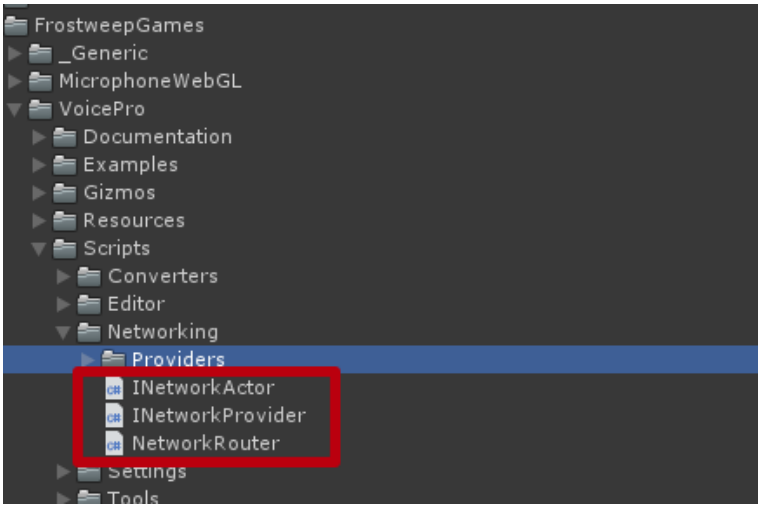
Here you could enable **Echo cancellation, Compressing of Data, Voice detection** settings, Speaker settings, etc.

The Speaker Config is affecting on all speakers. You could also set custom parent of each Speaker and set Spatial blend to 1 to make sound workable in 3D space.

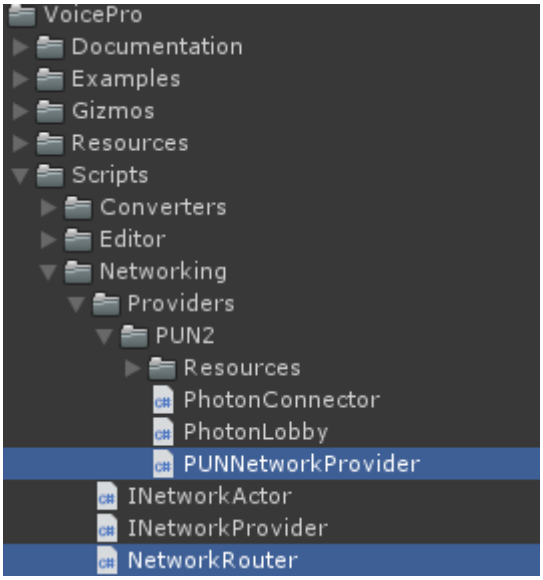
Advanced

We implemented PUN 2 and Mirror network solutions, however you also could implement your own.

We made few scripts that handles most of functionality for easy integration of different networks.



How to use them you could look at PUNNetworkProvider and NetworkRouter scripts:



For example - this is the list of function PUNNetworkProvider implements from interface:

```
/// <summary>
/// Photon PUN 2 Network Provider for data transmition
/// </summary>
Ссылок: 2
public class PUNNetworkProvider : INetworkProvider
{
    /// <summary>
    /// Code of network event that uses for voice data transition
    /// </summary>
    private const byte VoiceEventCode = 199;

    public event Action<INetworkActor, byte[]> NetworkDataReceivedEvent;

    private INetworkActor _networkActor;

    private GameObject _eventsHandler;

    Ссылок: 2
    public void Dispose()...

    Ссылок: 2
    public void Init(INetworkActor networkActor)...

    Ссылок: 2
    public void SendNetworkData(NetworkRouter.NetworkParameters parameters, byte[] bytes)...

    Ссылок: 2
    public string GetNetworkState()...

    Ссылок: 2
    public string GetConnectionToServer()...

    Ссылок: 2
    public string GetCurrentRoomName()...
```

Also there implemented PUNNetworkActor that uses for understanding of actors in network:

```
public class PUNNetworkActor : INetworkActor
{
    public string Id => Info.id;

    public string Name => Info.name;

    public bool IsAdmin { get; private set; }

    public NetworkActorInfo Info { get; private set; }

    [UnityEngine.Scripting.Preserve]
    public PUNNetworkActor(NetworkActorInfo info)...

    public void SetAdminStatus(bool status)...

    public override string ToString()...

    public static PUNNetworkActor FromString(string data)...
}
```

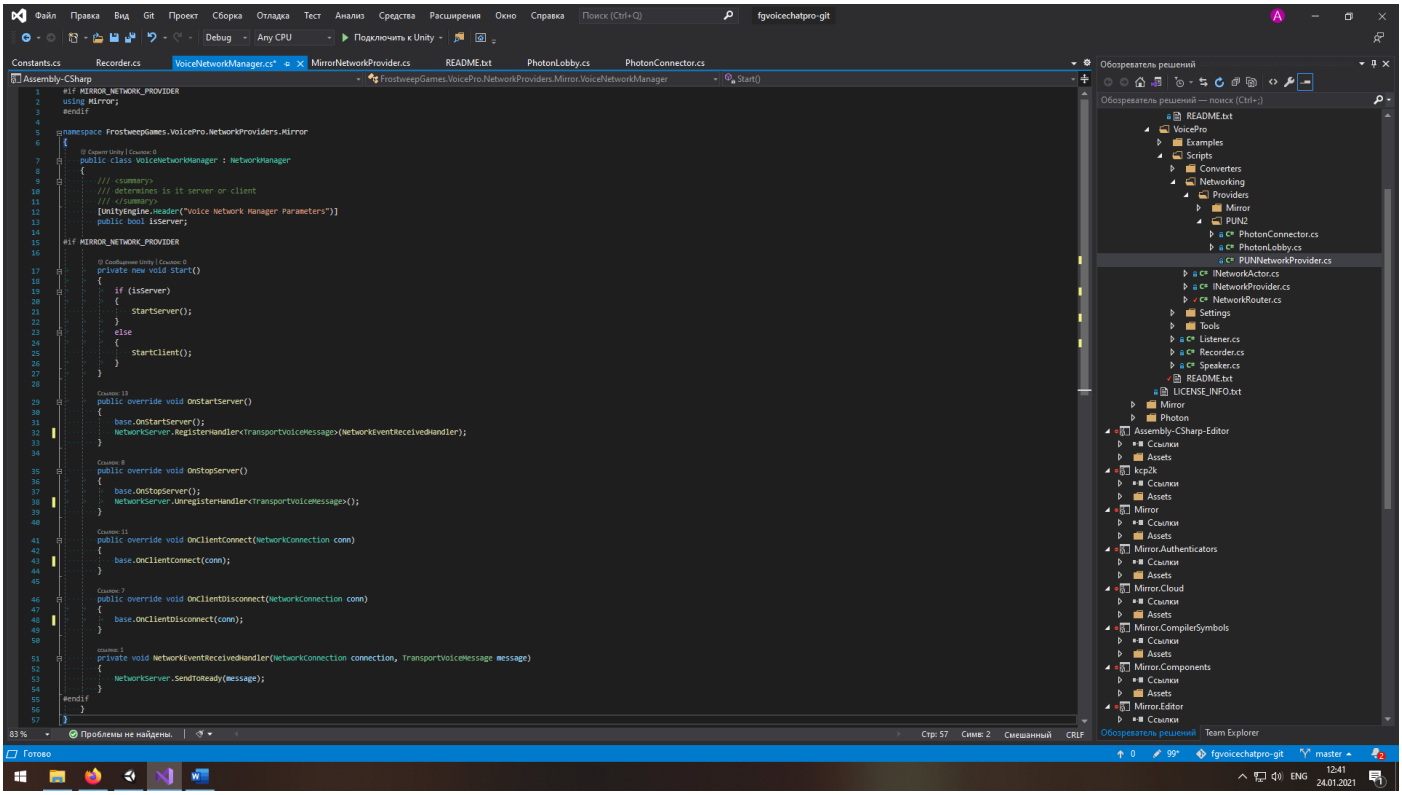
In NetworkRouter you have to implement registration of your network solution:

```
            switch (GeneralConfig.Config.networkProvider)
            {
                case GeneralConfig.NetworkProvider.PUN2_NETWORK_PROVIDER:
                {
#if PUN2_NETWORK_PROVIDER
                    _networkProvider = new PUNNetworkProvider();
                    networkActor = new PUNNetworkProvider.PUNNetworkActor(info);
#endif
                }
                break;
```

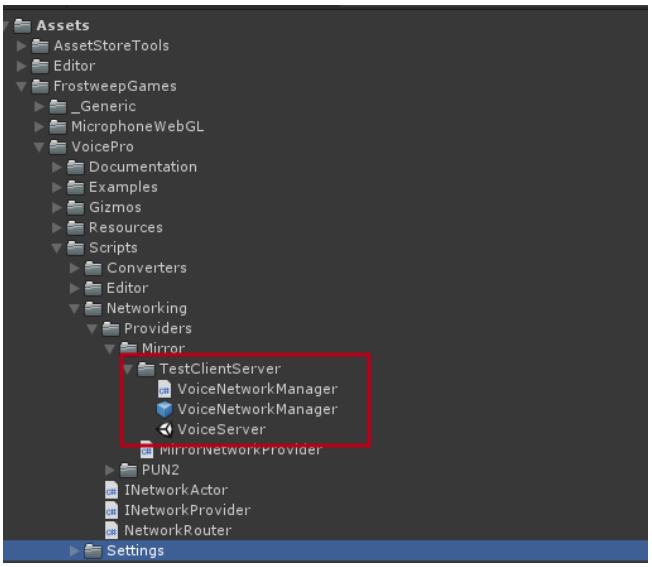
All connections to network written in other scripts such as lobby and connector.

How to setup Mirror Networking

For test purposes you could use our basic implementation of Mirror Server. We made VoiceNetworkManager which handles connection to server for clients and server starting functionality.



You can find this class and demo scene in:

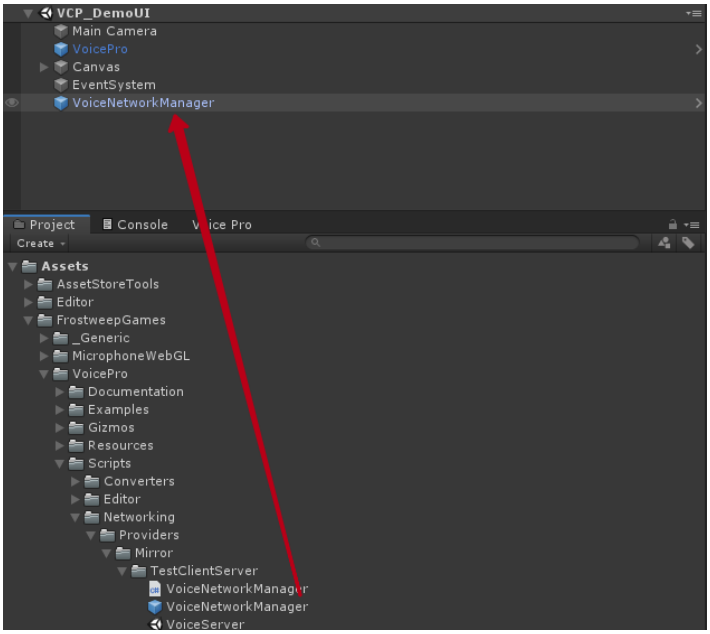


VoiceServer scene already includes prefab which contains VoiceNetworkManager configured for server

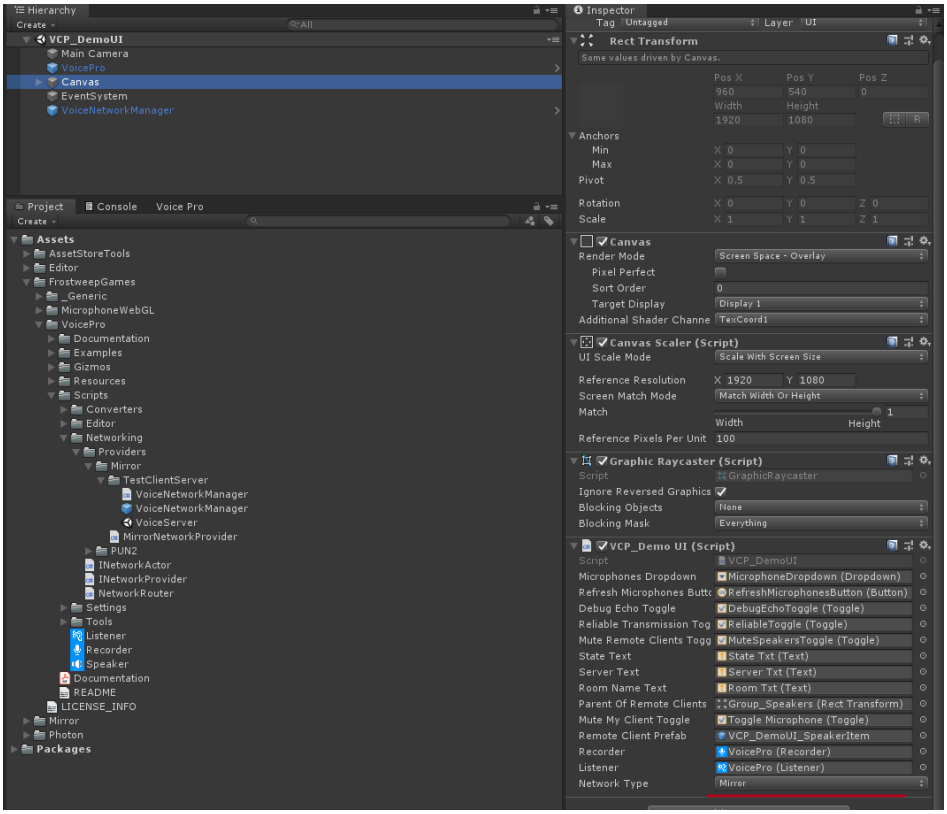
To start server build only VoiceServer scene and run it.

To make demo scene of Voice Pro workable with Mirror, you have to configure few things:

First of all you need top open demo scene and place configured prefab on the scene:



Then you need to select networking in Example script:



Now start the scene and enjoy!

API Reference available at: https://unitydemos.frostweepgames.com/assetstore/voiceprodemo/api/

WebGL demo: https://unitydemos.frostweepgames.com/assetstore/voiceprodemo/

Please read **README** file!