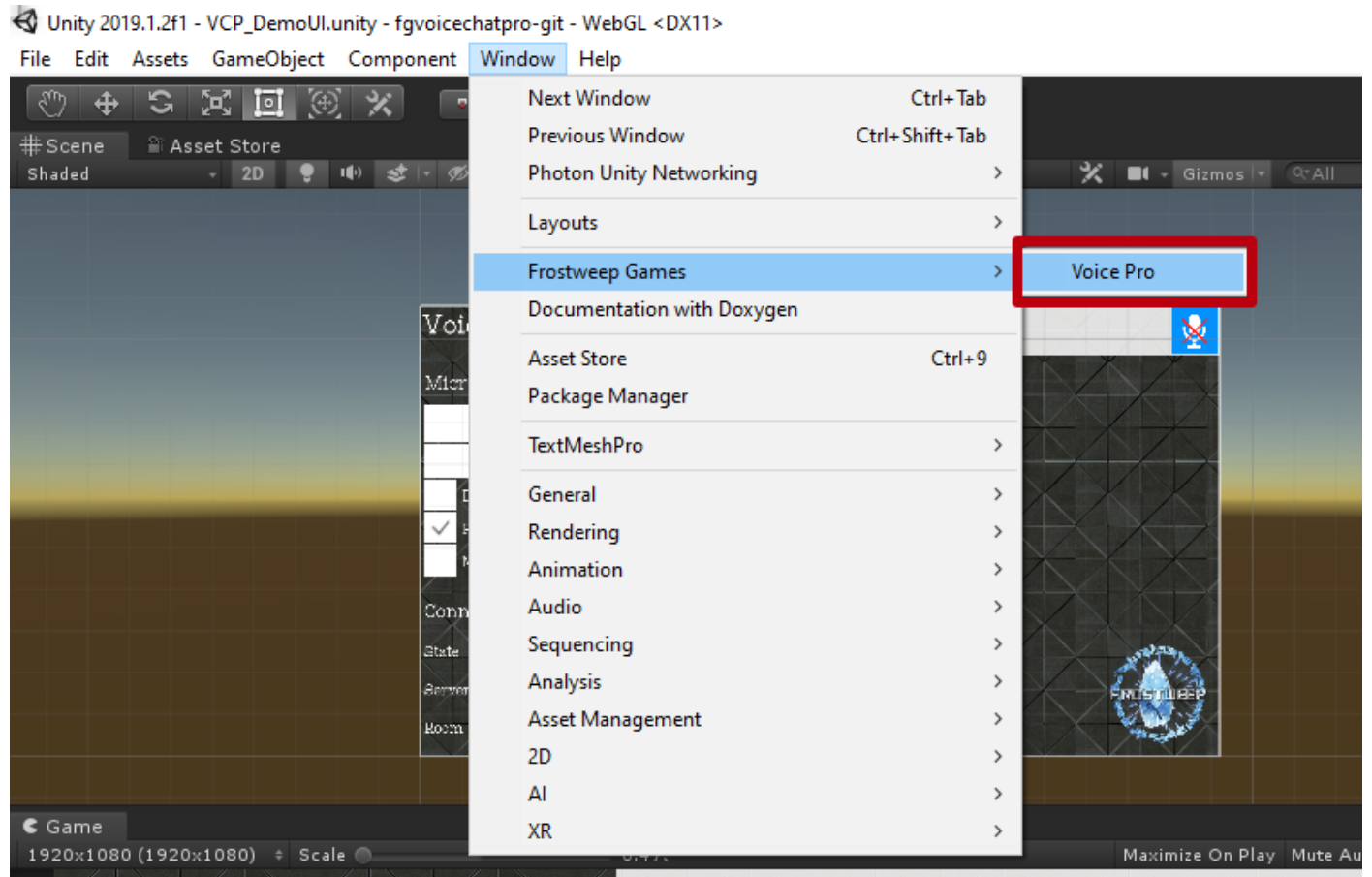
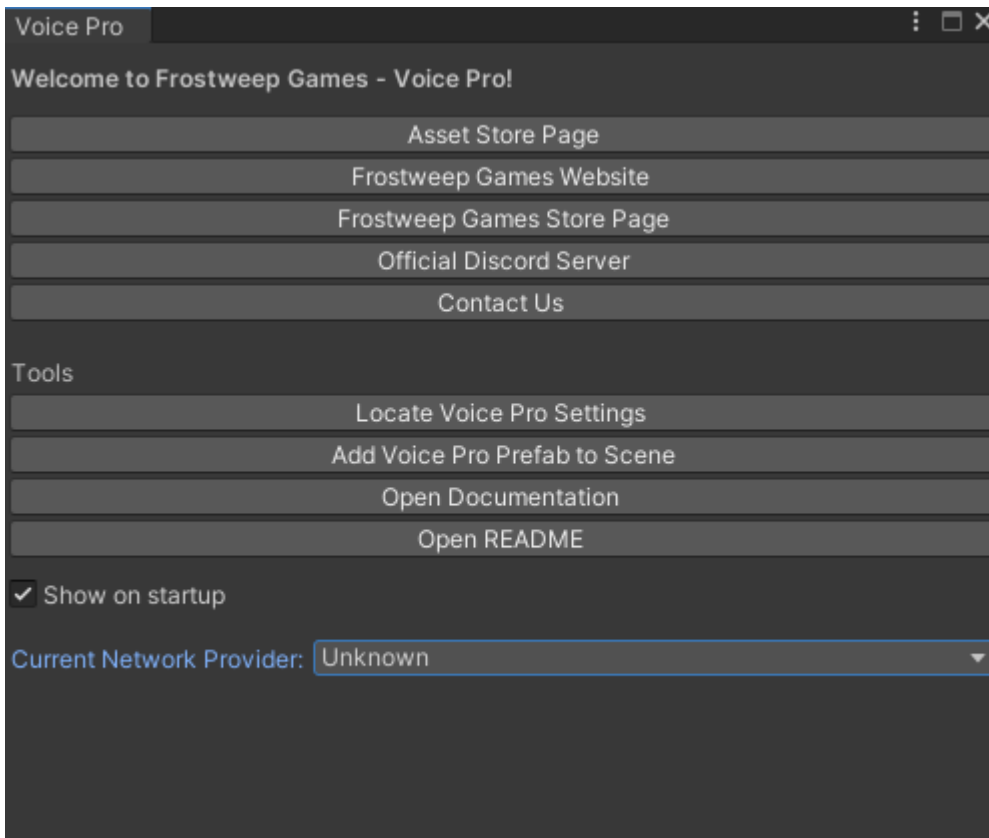


Voice Pro

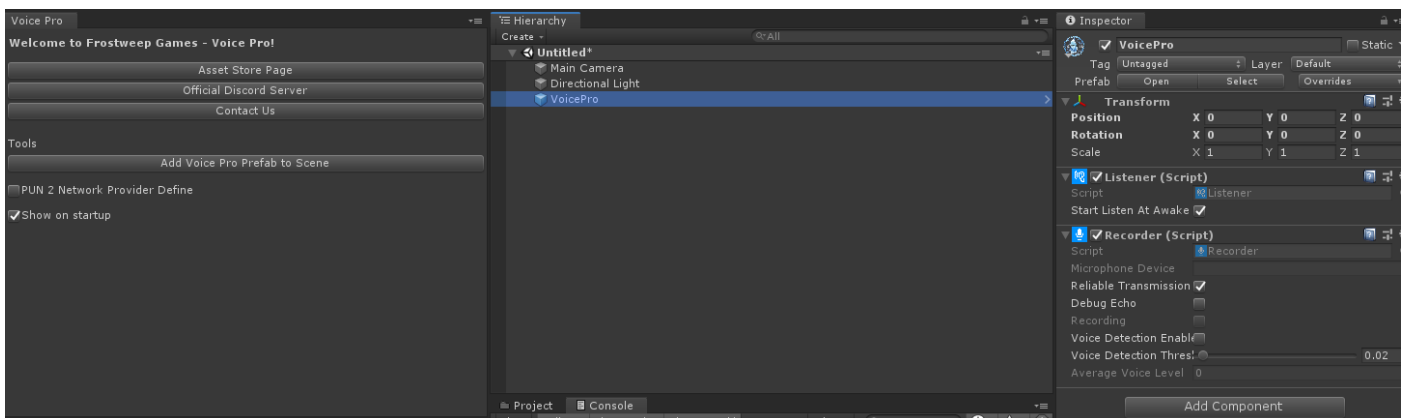
How to use

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



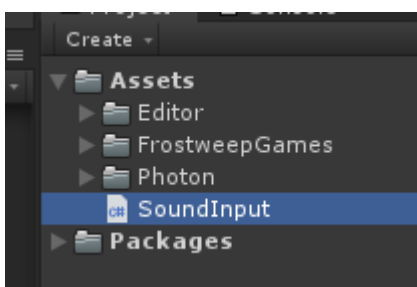


Create a new scene and click on *Add Voice Pro Prefab to Scene* button, it will create a new VoicePro object in scene.



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 X
Assembly-CSharp SoundInput Start()

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

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

```
Assembly-CSharp SoundInput

1 using FrostweepGames.VoicePro;
2 using UnityEngine;
3
4 [Script Unity | Ссылка: 0]
5 public class SoundInput : MonoBehaviour
6 {
7     public Recorder recorder;
8     public Listener listener;
9 }
10
11 // Start is called before the first frame update
12 [Сообщение Unity | Ссылка: 0]
13 void Start()
14 {
15 }
16
17 // Update is called once per frame
18 [Сообщение Unity | Ссылка: 0]
19 void Update()
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 function for start record and stop it. Lets name them accordingly.

```
3
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
12     // Сообщение Unity | Ссылка: 0
13     void Start()
14     {
15     }
16
17     // Update is called once per frame
18     // Сообщение Unity | Ссылка: 0
19     void Update()
20     {
21     }
22
23     // Сообщение: 0
24     public void StartRecord()
25     {
26     }
27
28     // Сообщение: 0
29     public void StopRecord()
30     {
31     }
32 }
33
```

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

```

@ Скрипт Unity | Ссылка: 0
public class SoundInput : MonoBehaviour
{
    public Recorder recorder;

    public Listener listener;

    // Start is called before the first frame update
    @ Сообщение Unity | Ссылка: 0
    void Start()
    {
        // ...
    }

    // Update is called once per frame
    @ Сообщение Unity | Ссылка: 0
    void Update()
    {
        // ...
    }

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

    @ Ссылка: 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 | Ссылка: 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:

```

public class SoundInput : MonoBehaviour
{
    public Recorder recorder;
    public Listener listener;
    public bool isRecording;

    // Сообщение Unity | Ссылка: 0
    void Start()
    {
        int id = Random.Range(0, 100);
        NetworkRouter.Instance.Register(id, "Speaker" + id, Enumerators.NetworkType.PUN2);
    }

    // Сообщение Unity | Ссылка: 0
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.R) && !isRecording)
        {
            StartRecord();
            isRecording = true;
        }
        else if (Input.GetKeyUp(KeyCode.R) && isRecording)
        {
            StopRecord();
            isRecording = false;
        }
    }

    // Ссылка: 1
    public void StartRecord()
    {
        recorder.StartRecord();
    }

    // Ссылка: 1
    public void StopRecord()
    {
        recorder.StopRecord();
    }
}

```

As it's a network solution you have to register networking. Currently our solution provides api for usage of PUN 2 network (**not** PUN Voice).

Lets import it from an asset store:

New to Unity? Checkout these top picks for all experience levels

unity Asset Store

Search for assets

Artem Frostweep ga...

Please rate and review this asset. Your honest review and rating will help other users who are deciding whether they should get this asset.

Write a review

photor²

UNITY NETWORKING

FREE

20 CCU

NEW VERSION

#1 Platform for

UNITY - Multiplayer

FAST. RELIABLE. SCALABLE.

PUN 2 - FREE

Exit Games

★★★★☆ 4 | 144 Reviews

FREE

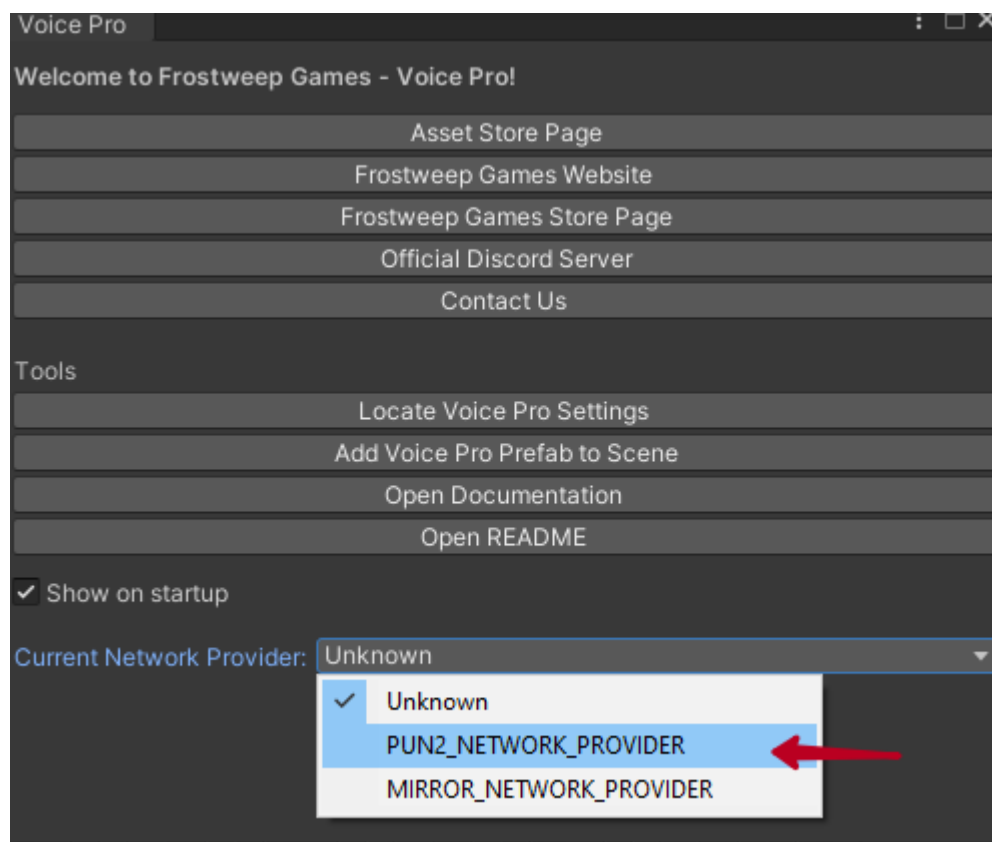
Import

Add to List

Share

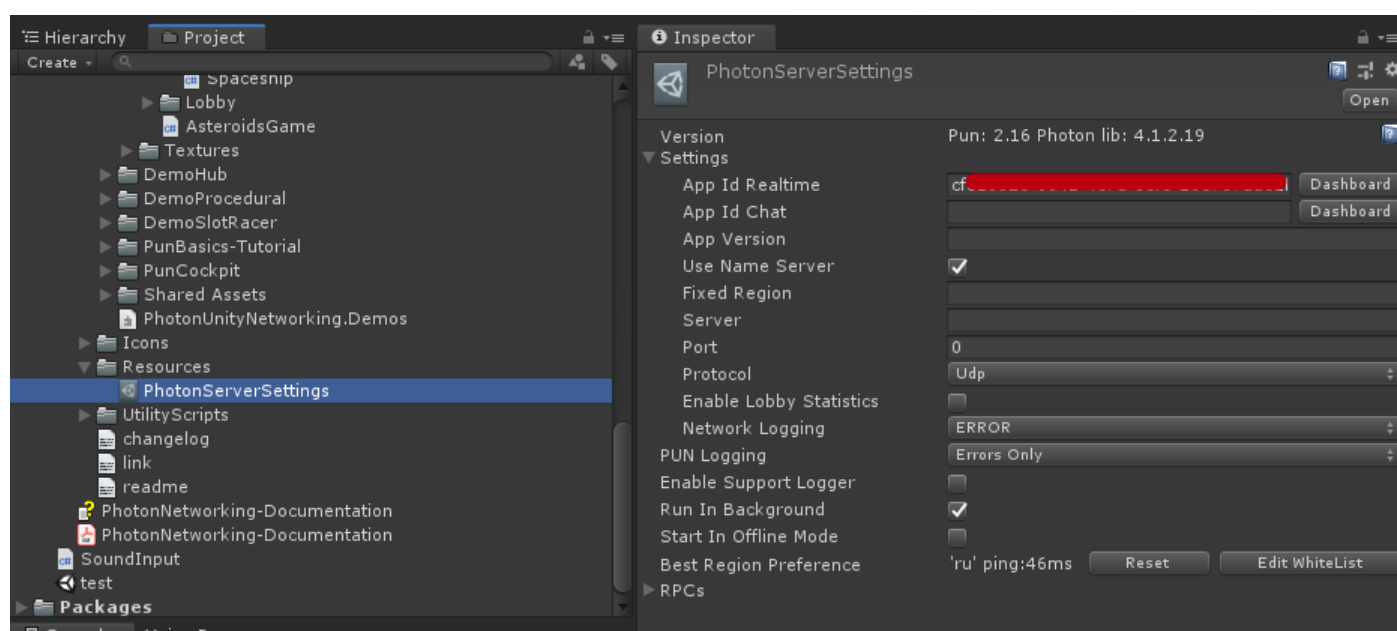
Frostweep Games, 2021

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)



Manage Test WebGL voice

[Dashboard](#)

App ID: cf-
XXXXXXXXXX

Properties

Name

Test WebGL voice

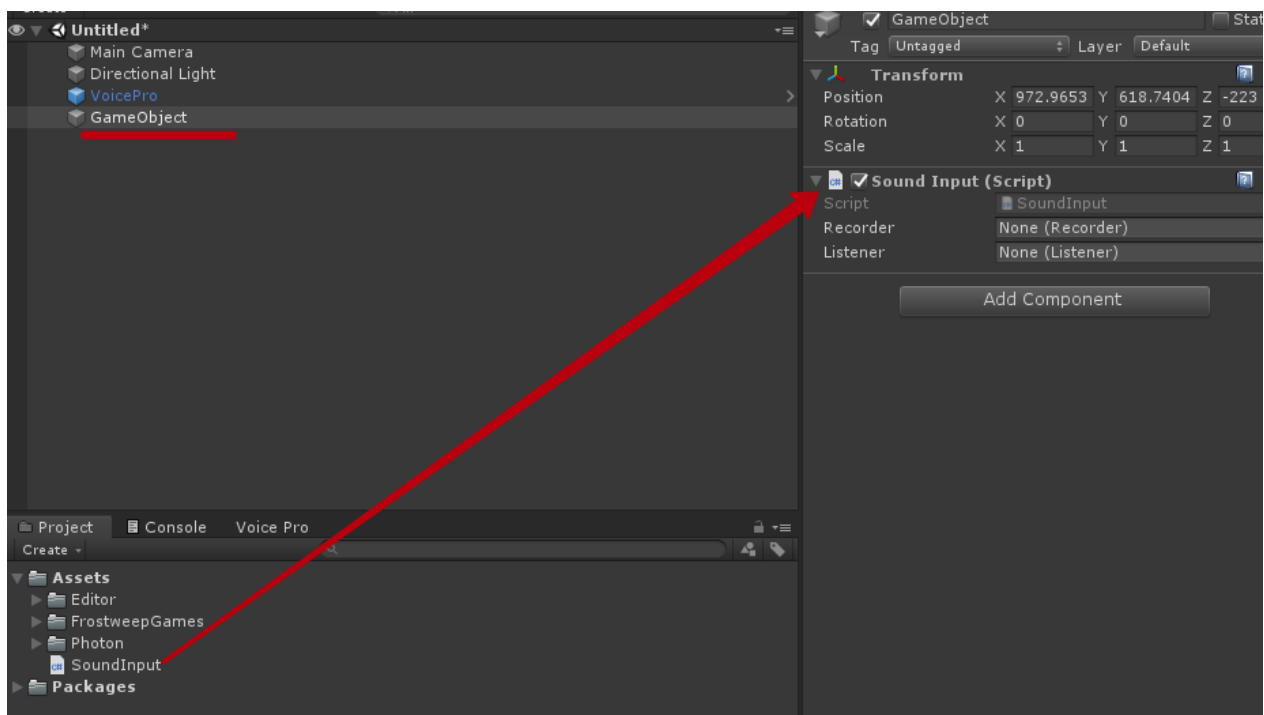
Url

Description

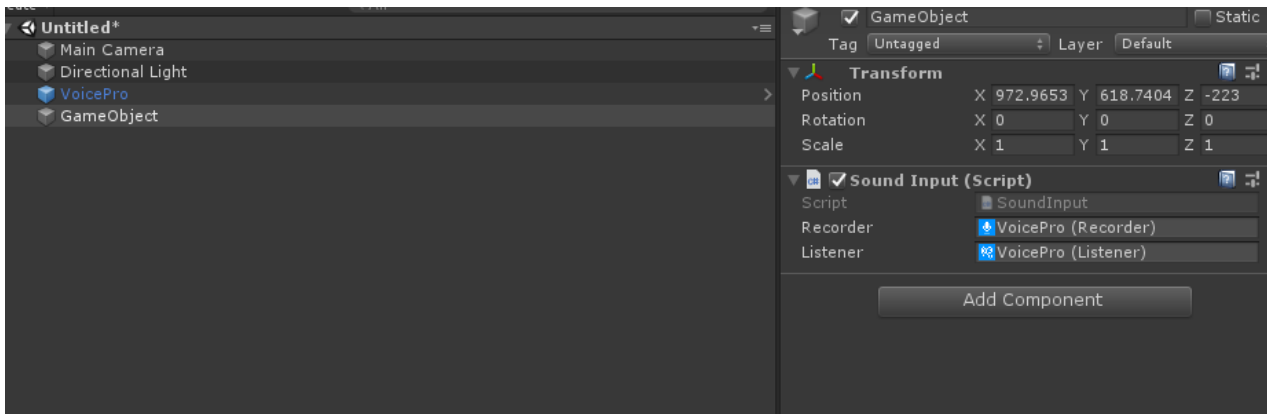
Concurrent Users

Subscription	0 CCU
One-Time	0 CCU
Coupon	0 CCU
Total	20 CCU CCU Burst is not allowed.

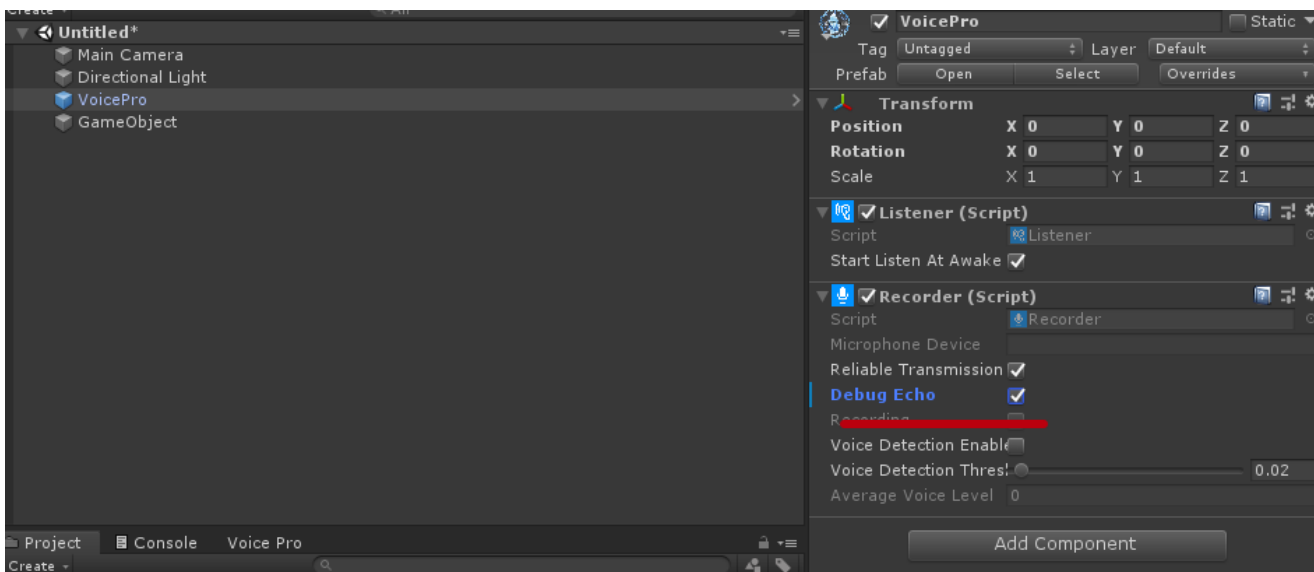
Last things that we have to do its create an empty object on created scene and attach SoundInput script we created on it(don't forget to save script before).



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

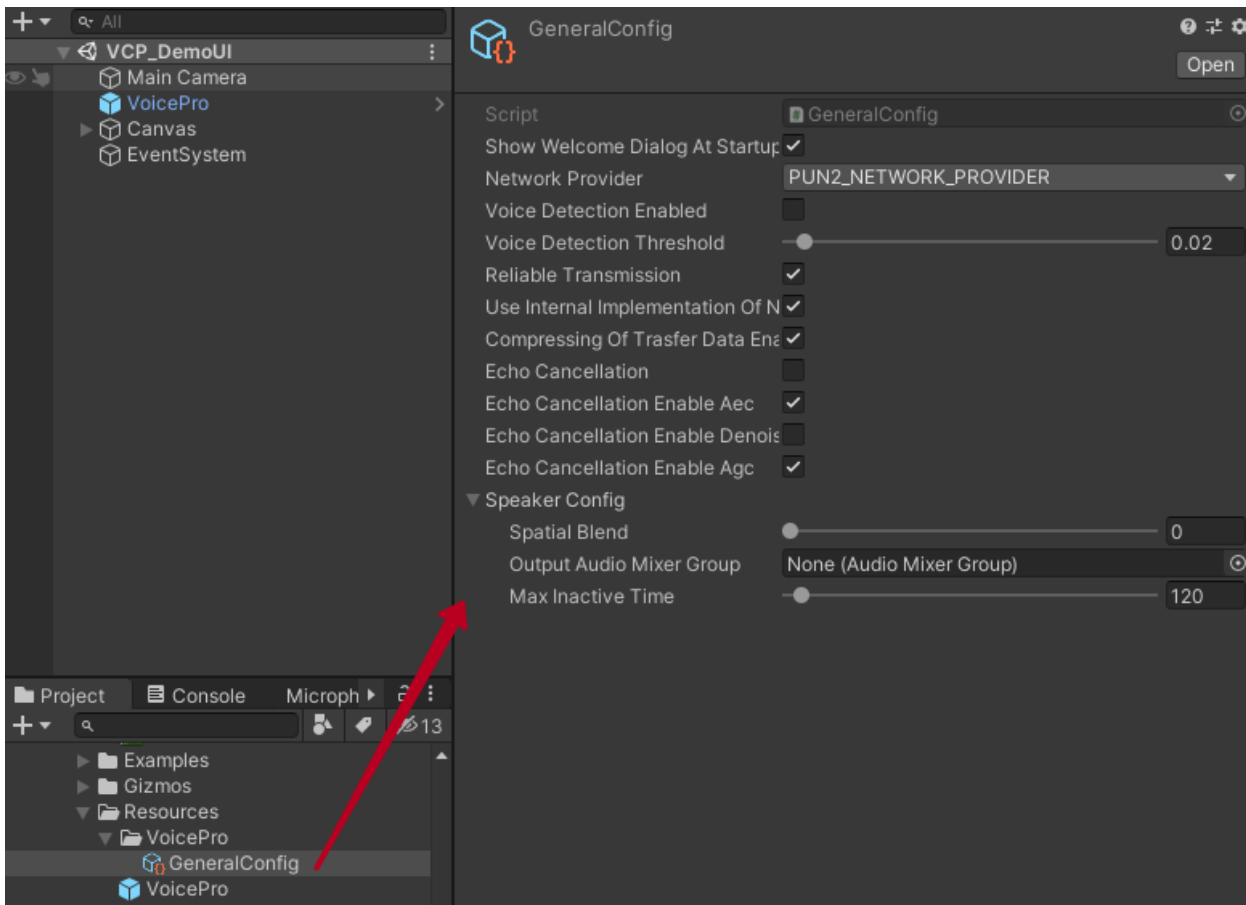


To make local test you could enable DebugEcho on Recorder component. Lets check DebugEcho checkbox:



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:



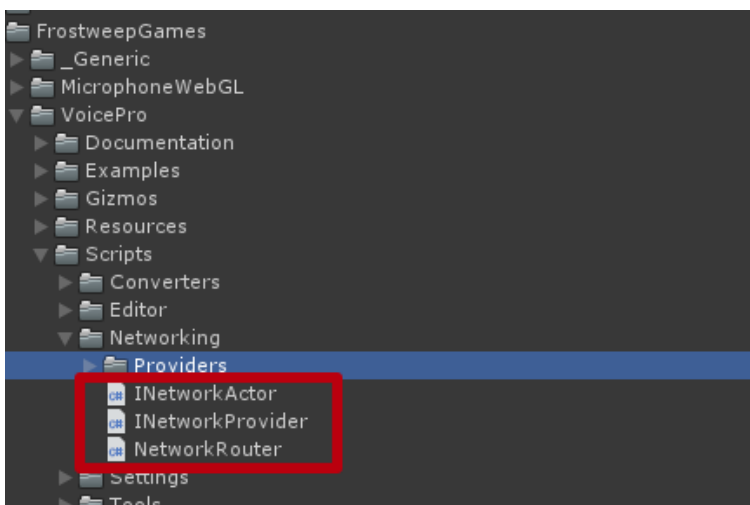
Here you could enable **Echo cancellation**, **Compressing of Data**, **Voice detection** settings, Speaker settings, etc.

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.

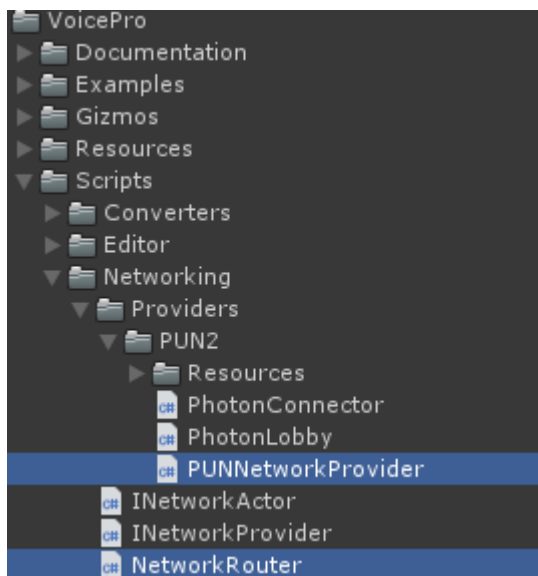
Advanced

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 transmission
</summary>
Ссылка: 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;

    Ссылка: 2
    public void Dispose()...

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

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

    Ссылка: 2
    public string GetNetworkState()...

    Ссылка: 2
    public string GetConnectionToServer()...

    Ссылка: 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:

```

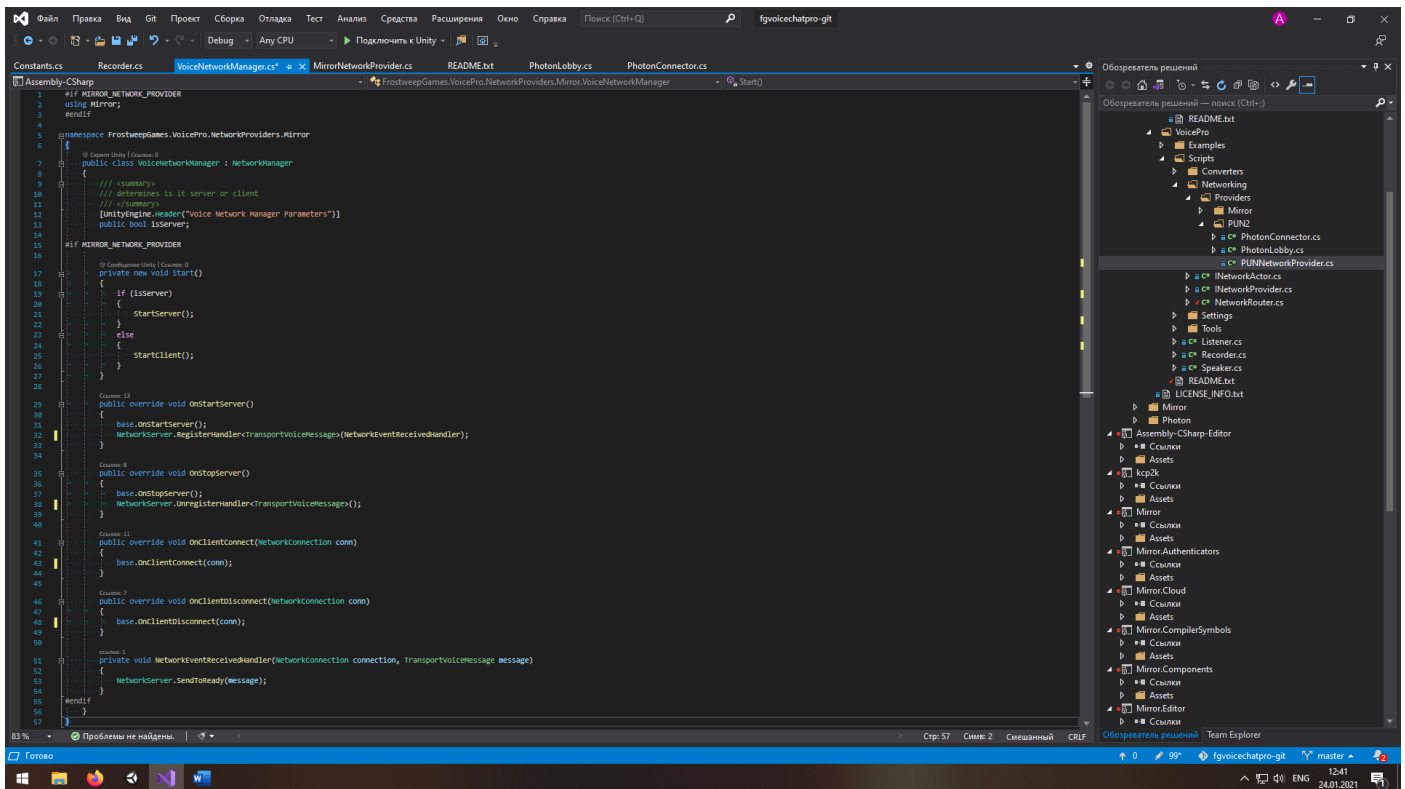
5 switch (GeneralConfig.Config.networkProvider)
6 {
7     case GeneralConfig.NetworkProvider.PUN2_NETWORK_PROVIDER:
8     {
9         #if PUN2_NETWORK_PROVIDER
10         {
11             _networkProvider = new PUNNetworkProvider();
12             networkActor = new PUNNetworkProvider.PUNNetworkActor(info);
13         }
14         #endif
15         break;
16     }
17 }

```

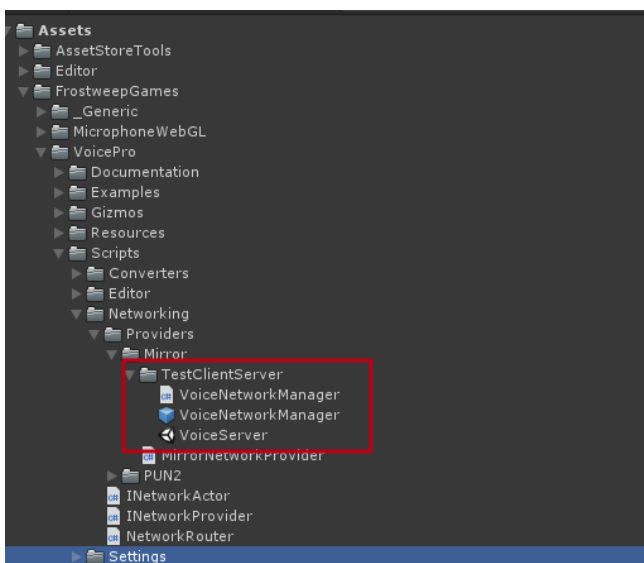
All connections to network written in other scripts such as lobby and connector.

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:

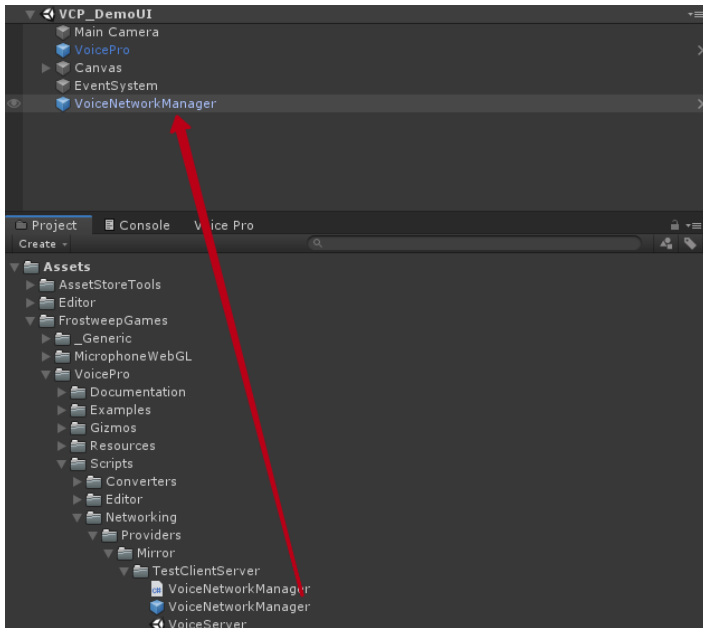


VoiceServer scene already includes prefab which contains VoiceNetworkManager configured for server

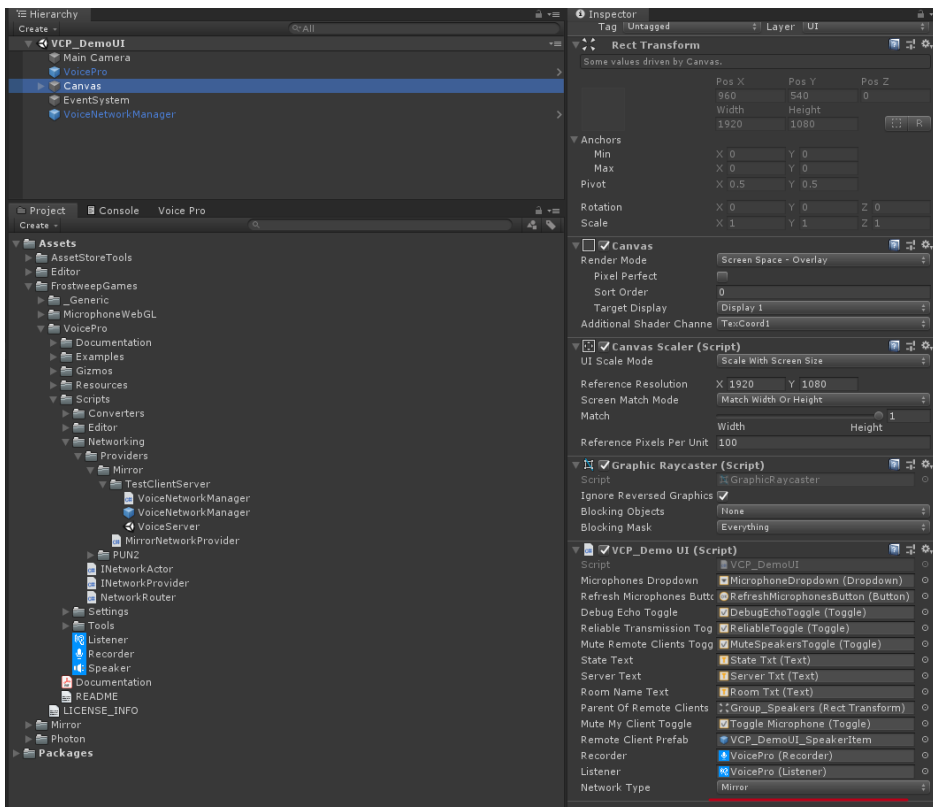
To start server build only VoiceServer scene and run it.

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!