

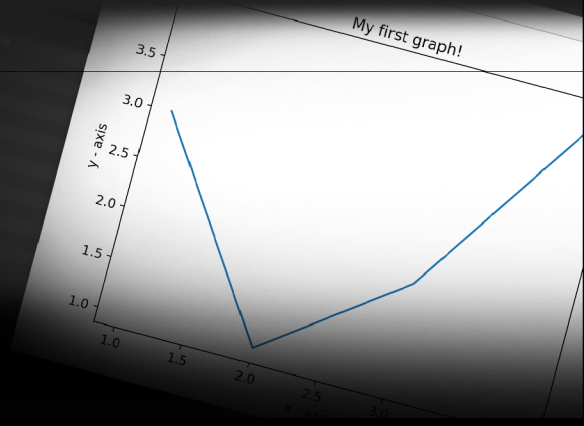
```

[-] Disconnected with ('127.0.0.1', 59169)
[+] Connected with ('127.0.0.1', 59172)
Listening...
('127.0.0.1', 59172) [CMD] python sum.py 1 2 3 4
Result:
10

[-] Disconnected with ('127.0.0.1', 59172)
[+] Connected with ('127.0.0.1', 59174)
Listening...
('127.0.0.1', 59174) [CMD] python sumarray.py 3,4,5,6
Result:
18.0

[-] Disconnected with ('127.0.0.1', 59174)

```



# UPython2



**ONLY ONE API YOU NEED**

```
pyChannel.Call("hello.py", Callback);
```

**DO NOT** forget to use the namespace

```
using DCXR;
```

- **Source code included.**



**ONLY ONE API YOU NEED**



**Introduction:**



**100 Credits:**

 [Installation:](#)

 [Validation: \(optional\)](#)

 [Tutorials:](#)

 [Multi Python Server:](#)

▶ [Please feel free to play the demo.](#)

 [Tips:](#)

[Pass variables:](#)

[Plot:](#)

 [Online Doc:](#)

[Discord](#)

[Unity Asset Store](#)

 [Github:](#)

 [About Me:](#)

## Introduction:

UPython2 upgraded from UPython. Essentially, it is a tool that bridges Unity and Python.

Unlike the first generation, this update applies multi-threading and asynchronous calling features, which makes UPython2 supports running multiple independent Python programs simultaneously.

***You can think of this tool as a tool for remotely invoking Python, which is very simple for the Unity client. You only need to call the API and wait for the data to return. But for the Python server, you need to prepare all the programs in advance to be called.***

UPython2 mainly consists of two parts: the Python server and the Unity client. This design allows the Unity client to access the Python application provided by the remote Python server. It can even access multiple remote Python servers at

the same time. Although, in most cases, we run the Python server and Unity client locally at the same time.

**UPython2 is a tool out of the box, as long as you have configured the Python environment.**

**In order to better integrate future products, the future version of the update will be packaged as a dll.**

## **100 Credits:**

If you like this asset or use it for any research project, please kindly give credit to this asset. Thanks.




## **Installation:**

Import it to your project.

***NOTE: DO NOT forget to download the [Python] folder. You NEED the Python server!!! You can put this folder anywhere. Download it from my Github repo.***


quincyhuang/MCMCFrameworks\_Public\_Repo

MCMCFrameworks\_Public\_Repo. Contribute to quincyhuang/MCMCFrameworks\_Public\_Repo development by creating an account on GitHub.

 [https://github.com/quincyhuang/MCMCFrameworks\\_Public\\_Repo](https://github.com/quincyhuang/MCMCFrameworks_Public_Repo)

quincyhuang/  
**MCMCFrameworks\_Publ...**

MCMCFrameworks\_Public\_Repo



1 Contributors

0 Issues

1 Stars

0 Forks



## **Validation: (opional)**

This step will help you to validate if the Python environment is set up correctly.

### 1. Start the Python server: (using cmd)

```
python server.py
```

You will see:

```
Server Started!  
Listening...
```

### 2. Start the Python:

```
python client.py
```

Then, you see the prompt:

```
CMD: python
```

Type:

```
hello.py
```

You will see:

```
CMD: python hello.py  
Rec:  
Hello World!  
Hi there.
```

### 3. Exit the server and client, you type:

```
exit()
```

When you exit from the server, you may have an error; you can ignore it. It was caused by Python 3.x, and it would not be a trouble to us.

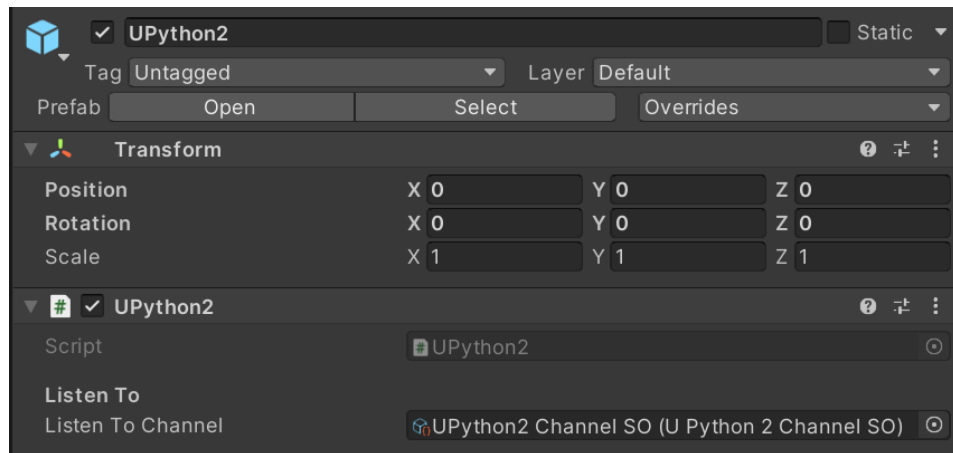


## Tutorials:

1. Start the Python server: (using cmd)

```
python server.py
```

2. In Unity, drag the prefab into the scene,  
**Assets/DCXR/UPython2/Prefabs/UPython2.prefab**

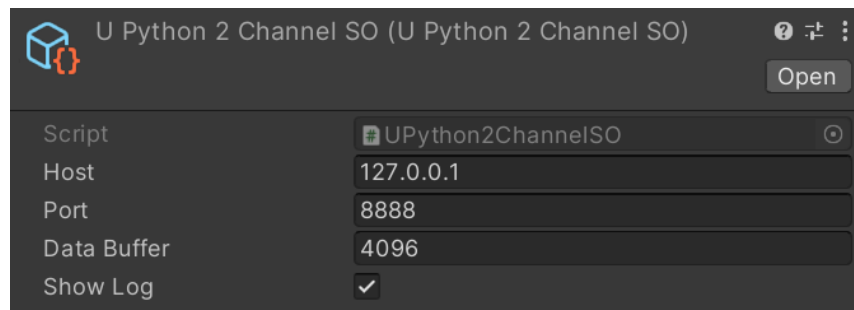


It attached a **UPython2** component which is the main component of the tool.

It requires a scriptable object that indicates the listening channel; this channel and UPython2 will handle all the remote Python server calls automatically. You do not need to worry about it.

**Just keep this game object [UPython2] in the scene all the time.**

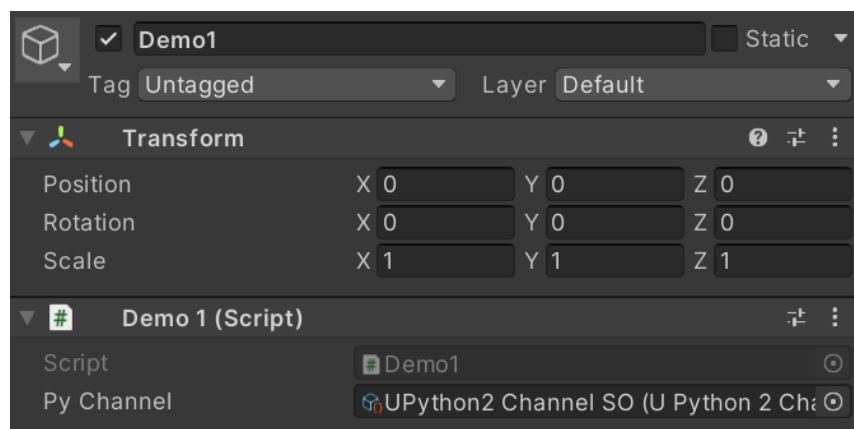
Double click to open the channel settings, **make sure you have the right Host IP and Port.** (see server.py)



You may free feel to adjust the size of the **Data Buffer** if needed.

3. Play the demo.

Select the game object **[Demo1]** in the scene, you will see:



Take a look at Demo1.cs, to get a better understanding of the tool.

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

namespace DCXR.Demo
{
    [HelpURL("")]
    public class Demo1 : MonoBehaviour
    {
```

```

[SerializeField] UPython2ChannelSO pyChannel;

void Callback(string data)
{
    Debug.Log($"[Rec:] {data}");
}

public void HelloWorld()
{
    pyChannel.Call("hello.py", Callback);
}

public void Sum()
{
    pyChannel.Call("sum.py 1 2 3 4", Callback);
}

public void SumArray()
{
    pyChannel.Call("sumarray.py 3,4,5,6", Callback);
}

public void Plot()
{
    pyChannel.Call("plot.py 1,2,3,4 3,1,2,4", null);
}
}
}

```

It requires a **UPython2ChannelSO** which is the one of you assigned to the **UPython2 game object**.

```
pyChannel.Call("hello.py", Callback);
```

Just one API you need, it is **Call()**.

The first parameter is cmd line, it indicates that which remotely command or Python Script you need to run (you can also put the variables follow by, just like what you do in the shell), and the second parameter is a callback function which will be called by system when any data comes from the Python Server.



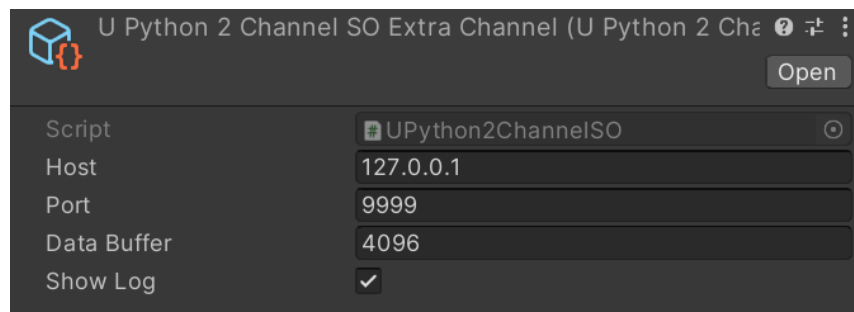
## Multi Python Server:

In most cases, you would not need this feature. Anyway, UPython2 supports this feature 100

1. Start the second Python Server:

```
python serverExtra.py
```

2. See the game object **[Demo2]** and **[UPython2 Extra Channel]** in Unity



You will see the second channel's settings that indicate it access to the second Python Server.

3. The Button **[Extra Server ML Demo]** on the screen which access to the second Python Server.

► Please feel free to play the demo.

Enjoy it and good luck. 



## Tips:

DO not forget to using the namespace

```
using DCXR;
```

## Pass variables:

Suppose you have tons of variables that need to be passed to the Python Server, and you run the Python Server local. In that case, you may save the variables into a file and let the Python Script read that file.

## Plot:

All plots will be pop-up on the Python Server. If you run Python Server remotely, you may want to convert the plots into a byte array and send it back to Unity Client. (Increase the Data Buffer if needed, both on the Python Server and Unity Client)

## Online Doc:

### UPython2

pyChannel.Call("hello.py", Callback); DO NOT forget to use the namespace UPyhon2 upgraded from UPython. Essentially, it is a tool that bridges Unity and Python. Unlike the first generation, this update applies multi-threading and asynchronous calling features, which makes UPyhon2 supports running <https://www.notion.so/UPython2-7431b13d3f0f4a41aa6fb6e16da782a3>

# Discord

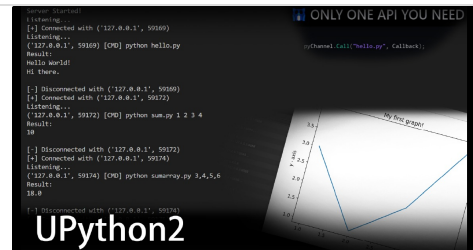
<https://discord.gg/NJnf52wt4R>

# Unity Asset Store

## UPython2 | Integration | Unity Asset Store

Use the UPython2 from Haikun Huang on your next project.  
Find this integration tool & more on the Unity Asset Store.


 <https://assetstore.unity.com/packages/tools/integration/upython2-193904>



## Github:

quincyhuang/MCMCFrameworks\_Public\_Repo

MCMCFrameworks\_Public\_Repo. Contribute to quincyhuang/MCMCFrameworks\_Public\_Repo development by creating an account on GitHub.

 [https://github.com/quincyhuang/MCMCFrameworks\\_Public\\_Repo](https://github.com/quincyhuang/MCMCFrameworks_Public_Repo)

quincyhuang/  
**MCMCFrameworks\_Publ...**

MCMCFrameworks\_Public\_Repo



1 Contributors 0 Issues 1 Stars 0 Forks



## About Me:

Haikun Huang

Reviewer, IEEE VR 2021 Conference Reviewer, CHI 2021  
Reviewer, Editorial 2020 Reviewer, Frontiers 2020 Reviewer,  
IEEE VR 2021 Reviewer, ACHI 2020 Reviewer, VRST 2020

<https://quincyhuang.github.io/Webpage/index.html>



## **Bug Report & Features Request**

[quincyhuang/MCMCFrameworks\\_Public\\_Repo](https://github.com/quincyhuang/MCMCFrameworks_Public_Repo)