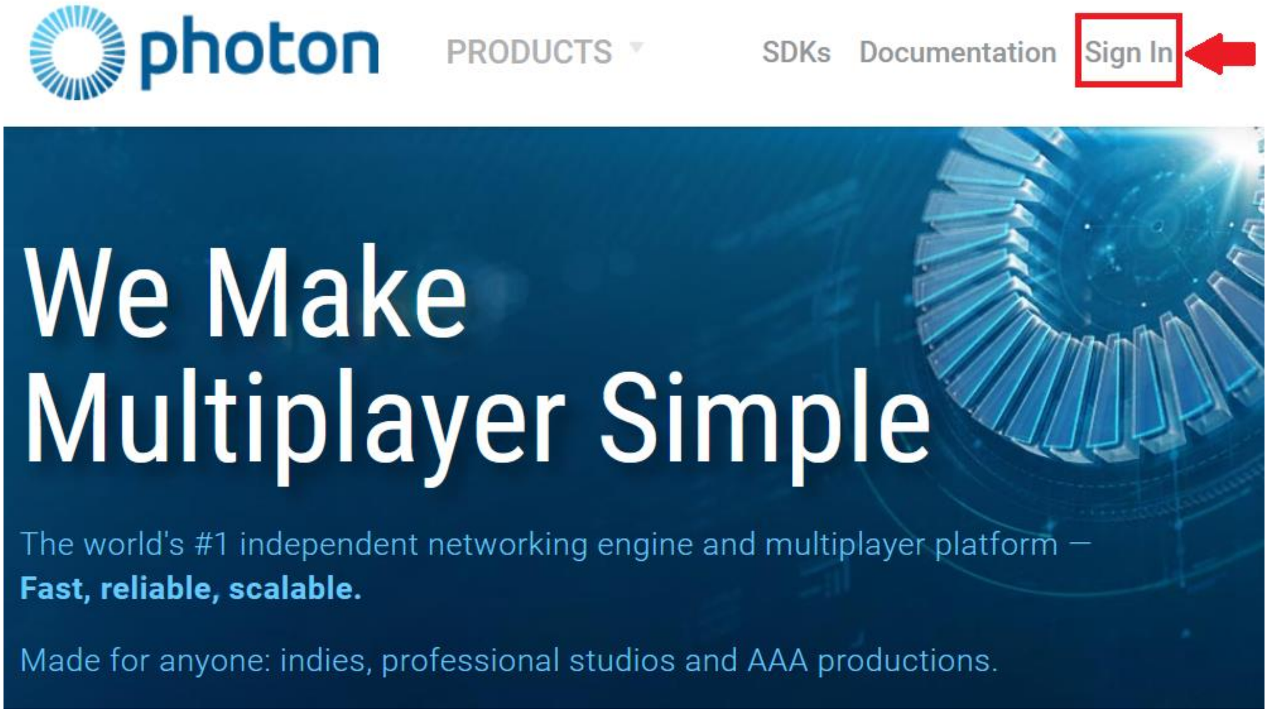
Photon Demo Guide

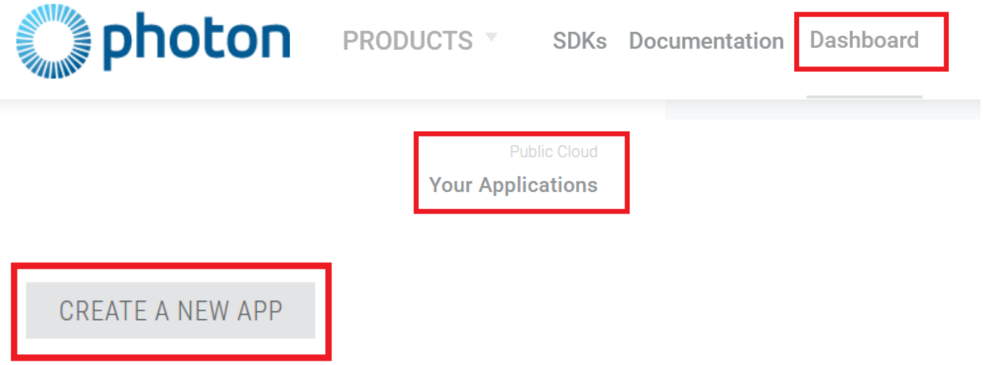
1 – Open your web browser, navigate to https://www.photonengine.com/ and Sign In (you will have to create an account if don't have one yet)



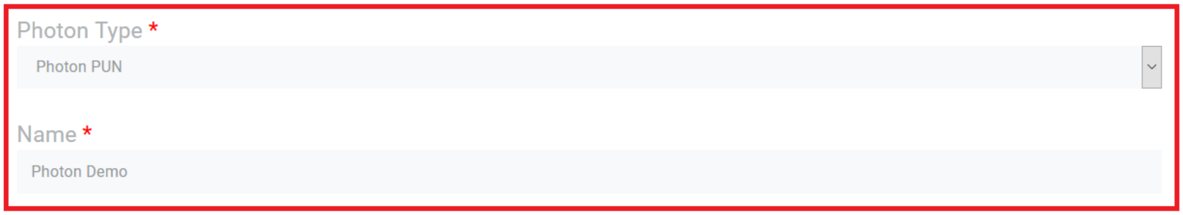
The world's #1 independent networking engine and multiplayer platform – **Fast, reliable, scalable.**

Made for anyone: indies, professional studios and AAA productions.

2 – Go to "Dashboard" => "Your Applications" => "CREATE A NEW APP"



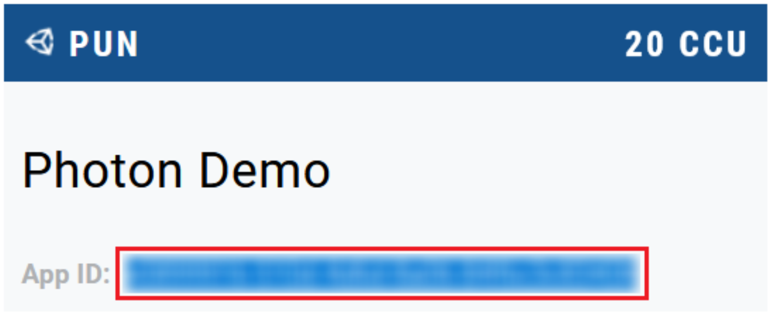
3 – For the "Photon Type" field, select "Photon PUN". Also give a name for your application. Finally click "CREATE". PUN stands for "Photon Unity Networking".



Photon Type *
Photon PUN

Name *
Photon Demo

4 – You will be redirected to your Dashboard. It will indicate “20 CCU”. It means your application is capable of handling up to 20 concurrent players. This is the default free tier Photon provides for any new application. Now your application was given an “App ID”. The page does not reveal the entire value. You have to click there so the value will show up. Copy the entire App ID value:

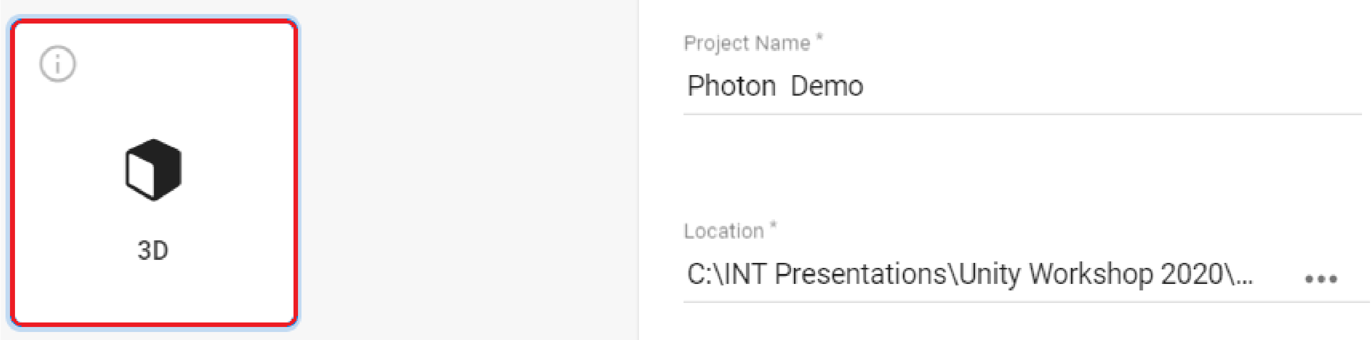


PUN 20 CCU

Photon Demo

App ID: [redacted]

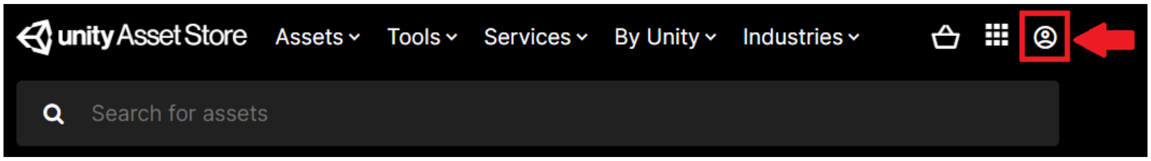
5 – Open up Unity and Create a new 3D project.



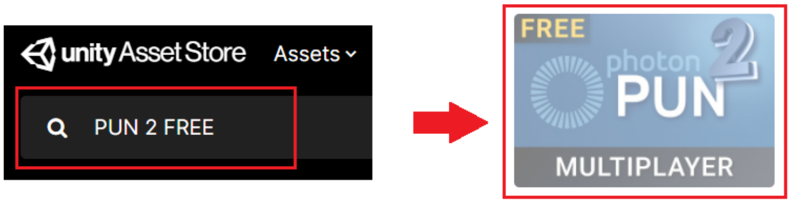
Project Name *
Photon Demo

Location *
C:\INT Presentations\Unity Workshop 2020\...

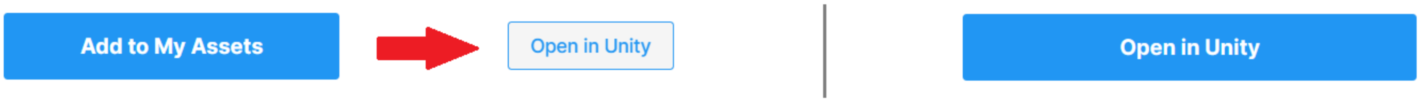
6 – Using your web browser, go to https://assetstore.unity.com/ and sign in with your Unity Account.



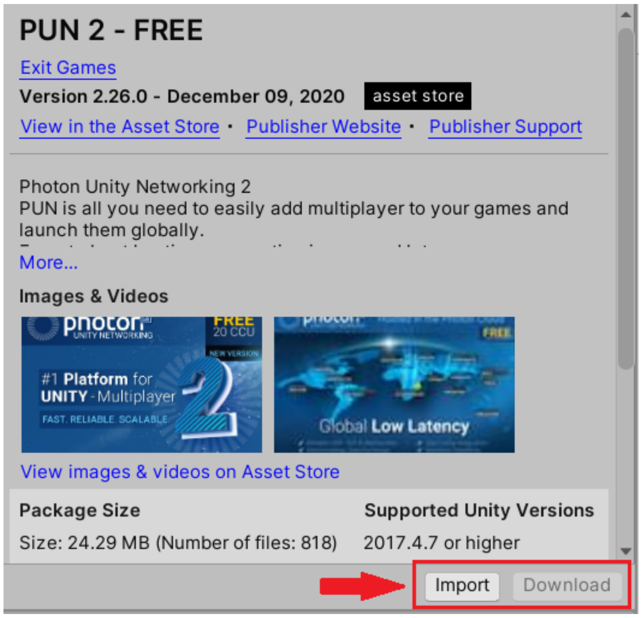
7 – Search for “PUN 2 free” and click the respective search result.



8 – When the “PUN 2 - FREE” page shows up, you will be presented with the possibility to open this item within the Package manager of Unity Editor. If this is your first time doing this step then click “Add to My Assets” => “Open in Unity”. Otherwise click directly “Open in Unity”. A dialog box will ask you to confirm and you may proceed.



9 – Back to Unity Editor, the Package Manager will show up. Proceed with the “Download” and “Import” processes. In the sequence a popup will list all files that will be imported into your project. Click the “Import” button of the popup.



PUN 2 - FREE

Exit Games

Version 2.26.0 - December 09, 2020 asset store

View in the Asset Store • Publisher Website • Publisher Support

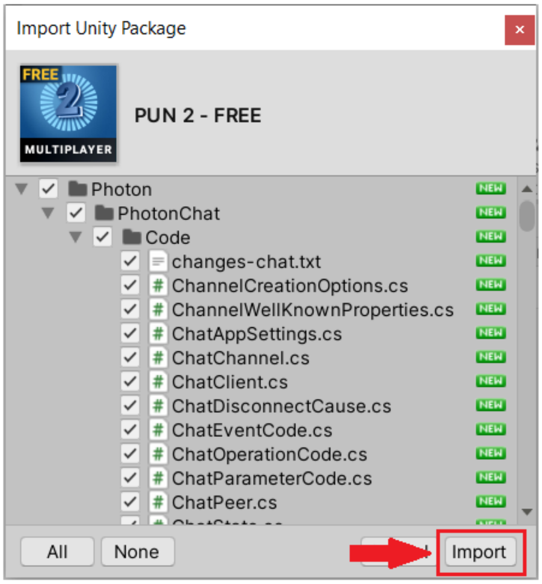
Photon Unity Networking 2
PUN is all you need to easily add multiplayer to your games and launch them globally.

More...

Images & Videos

View images & videos on Asset Store

Package Size	Supported Unity Versions
Size: 24.29 MB (Number of files: 818)	2017.4.7 or higher

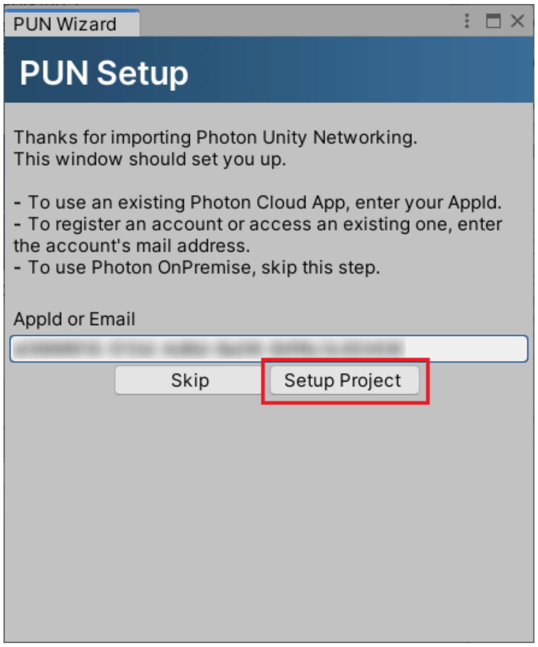


Import Unity Package

PUN 2 - FREE

```
▼ ☑ Photon                              NEW
  ▼ ☑ PhotonChat                        NEW
    ▼ ☑ Code                            NEW
        ☑ changes-chat.txt              NEW
        ☑ ChannelCreationOptions.cs     NEW
        ☑ ChannelWellKnownProperties.cs NEW
        ☑ ChatAppSettings.cs            NEW
        ☑ ChatChannel.cs                NEW
        ☑ ChatClient.cs                 NEW
        ☑ ChatDisconnectCause.cs        NEW
        ☑ ChatEventCode.cs              NEW
        ☑ ChatOperationCode.cs          NEW
        ☑ ChatParameterCode.cs          NEW
        ☑ ChatPeer.cs                   NEW
```

10 – After the import process is done, the “PUN Wizard” window will ask you the App ID you created previously. Paste the App ID value into the text field and click “Setup Project” and then close this window. Click on the Scene window tab.



PUN Wizard

PUN Setup

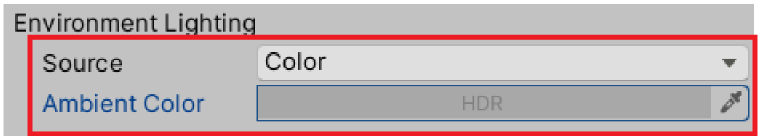
Thanks for importing Photon Unity Networking.
This window should set you up.

- To use an existing Photon Cloud App, enter your AppId.
- To register an account or access an existing one, enter the account's mail address.
- To use Photon OnPremise, skip this step.

AppId or Email

[redacted]

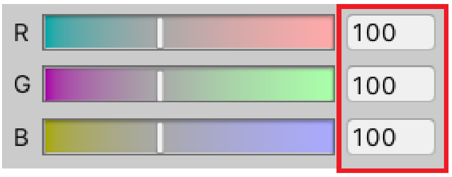
11 – Go to the menu “Window” => “Rendering” => “Lighting”. In the Lighting window, click “Environment” tab. Under “Environment Lighting”, set “Source” property to “Color”. Click the “Ambient Color” property. In the HDR Color window, set the R, G, and B values to “**100**”. Close the HDR Color and Lighting windows.



Environment Lighting

Source: Color

Ambient Color: HDR



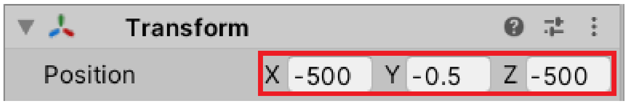
R 100

G 100

B 100

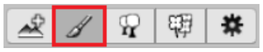
12 – In the Project window, collapse the Photon folder if it is expanded. Right click Assets folder and create a folder called “Textures”. Using your web browser, go to http://bit.ly/workshoptexture (https://i.ibb.co/d7RhDGx/Ground.png), download this image and save in the Textures folder you just created.

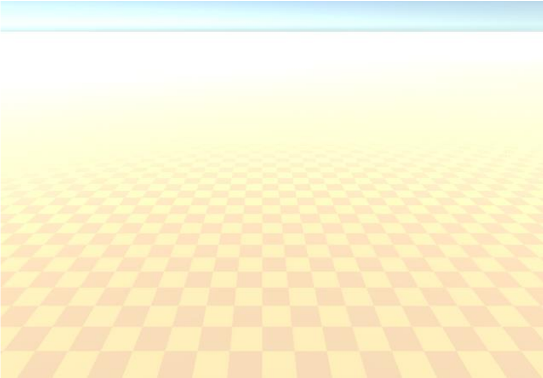
13 – In the main menu, click “GameObject” => “3D Object” => “Terrain”. Set the XYZ position to **-500**, **-0.5**, and **-500** respectively.



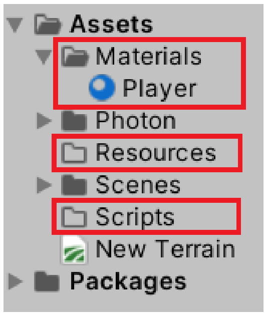
Transform

Position X -500 Y -0.5 Z -500

14 – Click on the “Paint Terrain” icon  => ⚙ Edit Terrain Layers... => “Create Layer” menu. In the search bar type “**Ground**”, select the texture and close the popup window. The terrain will look like this:



15 – In the Project window, under “Assets”, create the 3 folders: “**Materials**”, “**Resources**”, and “**Scripts**”. To create folder, right click “Assets” => “Create” => “Folder”. Right click the Materials folder and create a material called “**Player**”.



Assets
Materials
Player
Photon
Resources
Scenes
Scripts
New Terrain
Packages

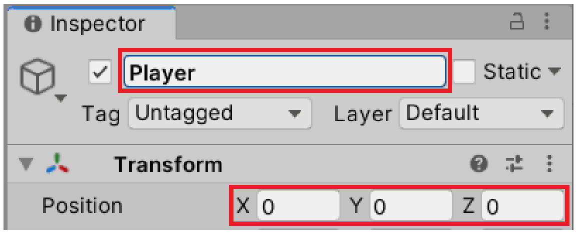
16 – Right click the Scripts folder, click “Create” => “C# Script”, and name it as “**CharacterMovement**”. Repeat the process and create the scripts “**Player**”, “**UIManager**”, and “**RoomManager**”. VERY IMPORTANT**: DON’T FORGET TO CREATE THESE FOUR SCRIPTS.**

17 – Open the CharacterMovement script, delete all lines, and paste the following code:

```
using Photon.Pun;
using UnityEngine;

public class CharacterMovement : MonoBehaviour
```

```
{
    [SerializeField]
    private float characterSpeed = 8;

    [SerializeField]
    private float jumpSpeed = 12;

    [SerializeField]
    private float gravity = 20;

    private CharacterController characterController;
    private Vector3 inputVec = new Vector3();
    private PhotonView view;

    private void Awake()
    {
        characterController = GetComponent<CharacterController>();
    }

    private void Start()
    {
        view = GetComponent<PhotonView>();
        if (view.IsMine || !PhotonNetwork.InRoom)
        {
            AttachCamera();
        }
    }

    internal void AttachCamera()
    {
        GameObject.Find(“Main Camera”).transform.parent = transform;
    }

    private void Update()
    {
        UpdatePositionByInput();
    }

    private void UpdatePositionByInput()
    {
        if (!view.IsMine && PhotonNetwork.IsConnected)
            return;

        inputVec.x = Input.GetAxis("Horizontal") * characterSpeed;
        inputVec.z = Input.GetAxis("Vertical") * characterSpeed;

        if (characterController.isGrounded && Input.GetButtonDown("Jump"))
        {
            inputVec.y = jumpSpeed;
        }

        inputVec.y -= gravity * Time.deltaTime;
        characterController.Move(inputVec * Time.deltaTime);
    }
}
```

18 – Back to Unity Editor, in the main menu, click “GameObject” => “3D Object” => “Cube”. Rename to Cube as “Player” and set the XYZ position to **0**, **0** and **0** respectively.



Inspector

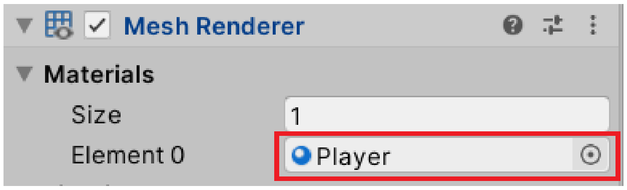
☑ Player □ Static

Tag Untagged Layer Default

Transform

Position X 0 Y 0 Z 0

19 – Still in the Inspector window, drag and drop the “**Player**” material into the “**Element 0**” property for the Mesh Renderer component.



☑ Mesh Renderer

Materials

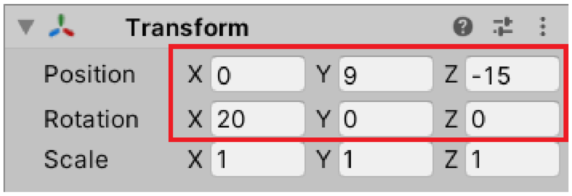
Size 1

Element 0 Player

20 – Still in the Inspector window, scroll down to the bottom. By clicking the “Add Component” button, search and add the following components: “**Player**”, “**Character Controller**”, “**Character Movement**”, “**Photon View**”, “**Photon Transform View**”.

21 – For the Character Controller component, set the Skin Width property value to “**0**”.

22 – Select the Game window tab. Set the following for the Main Camera:

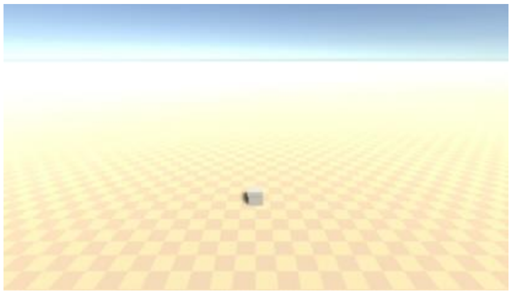


Transform

Position X 0 Y 9 Z -15

Rotation X 20 Y 0 Z 0

Scale X 1 Y 1 Z 1

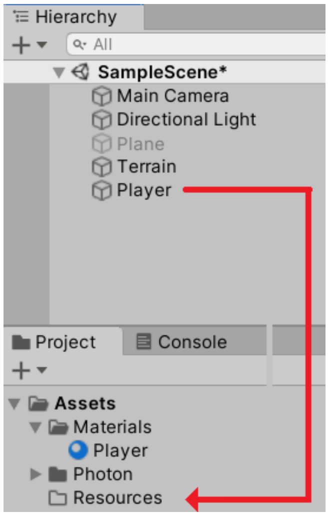


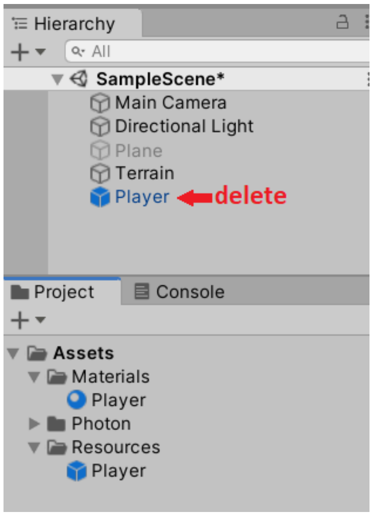
23 – Press the Play button in the Unity Editor and test the character movement using arrow or wasd keys.

24 – Open the Player script, delete all lines, and paste the following code:

```
using Photon.Pun;
using UnityEngine;
using UnityEngine.UI;

public class Player : MonoBehaviour
{
    private PhotonView view;
    private TextMesh nicknameLabel;

    private void Awake()
    {
        view = GetComponent<PhotonView>();
    }

    private void Start()
    {
        AddNicknameLabel();
        RetrieveInitialColor();
    }

    private void RetrieveInitialColor()
    {
        string colorName = view.Owner.CustomProperties["ColorName"].ToString();
        UpdateColor(colorName);
    }

    private void AddNicknameLabel()
    {
        GameObject nicknameGameObject = new GameObject("NicknameLabel");
        nicknameLabel = nicknameGameObject.AddComponent<TextMesh>();
        nicknameLabel.text = GetComponent<PhotonView>().Owner.NickName;
        nicknameLabel.color = Color.grey;
        nicknameLabel.fontStyle = FontStyle.Bold;
        nicknameLabel.alignment = TextAlignment.Center;
        nicknameLabel.anchor = TextAnchor.MiddleCenter;
        nicknameLabel.characterSize = 0.065f;
        nicknameLabel.fontSize = 120;
        nicknameGameObject.transform.parent = transform;
        nicknameGameObject.transform.localPosition = new Vector3(0, 1.5f, 0);
    }

    [PunRPC]
    internal void UpdateNickname(string nickname)
    {
        view.Owner.NickName = GameObject.Find("InputField_Nickname").GetComponent<InputField>().text;
        nicknameLabel.text = nickname;

        if (view.IsMine)
        {
            view.RPC("UpdateNickname", RpcTarget.Others, nickname);
        }
    }

    [PunRPC]
    internal void UpdateColor(string colorName)
    {
        Color color;
        ColorUtility.TryParseHtmlString(colorName, out color);
        gameObject.GetComponent<Renderer>().material.color = color;

        if (view.IsMine)
        {
            PhotonNetwork.LocalPlayer.CustomProperties["ColorName"] = colorName;
            PhotonNetwork.LocalPlayer.SetCustomProperties(PhotonNetwork.LocalPlayer.CustomProperties);
            view.RPC("UpdateColor", RpcTarget.Others, colorName);
        }
    }
}
```

25 – Drag the Player game object from the Hierarchy window and drop inside the Resources folder in the Project window. It will create the Player prefab. If the prefab was successfully created, it will be turned to blue. Delete Player game object in the Hierarchy window.





26– In the main menu, click “GameObject” => “Create Empty”. Name it as “**RoomManager**”. Add the **RoomManager** component to this game object.

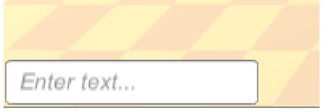
27 – Open the RoomManager script, delete all lines, and paste the following code:

```
using Photon.Pun;
using Photon.Realtime;
using UnityEngine;
using UnityEngine.UI;

public class RoomManager : MonoBehaviourPunCallbacks
{
    private void Start()
    {
        Connect();
    }

    private void Connect()
    {
        Display("Connecting...");
        PhotonNetwork.ConnectUsingSettings();
    }

    public override void OnConnectedToMaster()
    {
        PhotonNetwork.JoinLobby();
    }

    public override void OnJoinedLobby()
    {
        if (!PhotonNetwork.InRoom)
        {
            Display("Joined Lobby");
            SetupMyPlayer();
            var joinRoomParams = new OpJoinRandomRoomParams();
            var createRoomParams = new EnterRoomParams();
            PhotonNetwork.NetworkingClient.OpJoinRandomOrCreateRoom(joinRoomParams, createRoomParams);
        }
    }

    private void SetupMyPlayer()
    {
        string nickname = GameObject.Find("InputField_Nickname").GetComponent<InputField>().text;
        Dropdown ddColor = GameObject.Find("Dropdown_Color").GetComponent<Dropdown>();
        string colorName =  ddColor.options[ddColor.value].text;
        PhotonNetwork.NickName = nickname;
        PhotonNetwork.LocalPlayer.CustomProperties["ColorName"] = colorName;
    }

    public override void OnJoinedRoom()
    {
        SpawnMyPlayer();
    }

    public override void OnCreateRoomFailed(short returnCode, string message)
    {
        Display("Room Creation Failed: " + message);
    }

    public override void OnLeftRoom()
    {
        Display("Left Room");
    }

    private void Display(string info)
    {
        Debug.Log(info);
    }

    private void SpawnMyPlayer()
    {
        var player = PhotonNetwork.Instantiate("Player", Vector3.zero, Quaternion.identity);
        player.GetComponent<CharacterMovement>().AttachCamera();
        int rndX = Random.Range(-5, 5);
        int rndZ = Random.Range(-5, 5);
        player.transform.position = new Vector3(rndX, 0, rndZ);
        player.tag = "Player";
    }
}
```

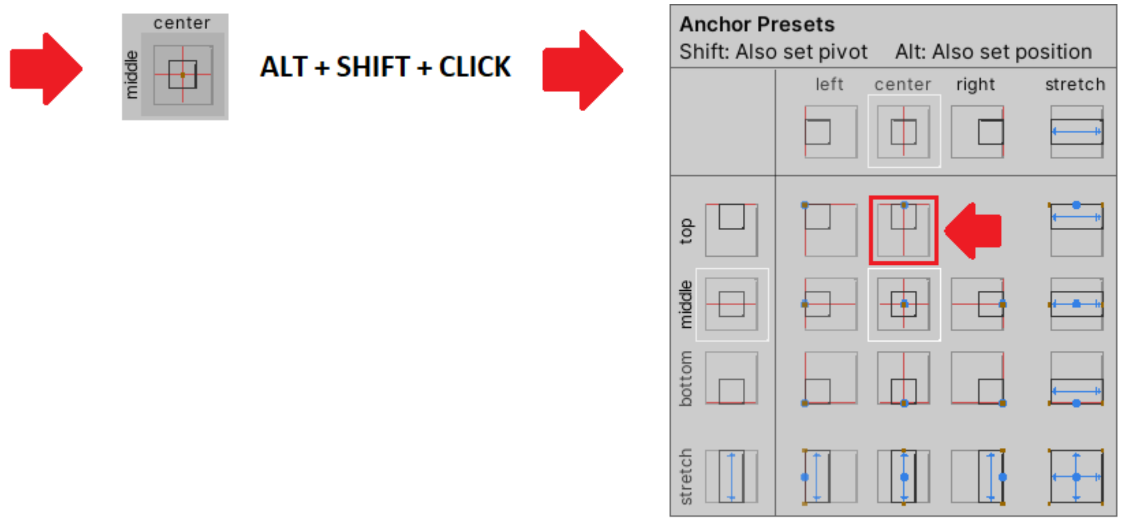
28 – Back to Unity Editor, open the UIManager script, delete all lines, and paste the following code:

```
using UnityEngine;
using UnityEngine.UI;

public class UIManager : MonoBehaviour
{
    public void OnNicknameChange(string nickname)
    {
        GameObject.FindGameObjectWithTag("Player").GetComponent<Player>().UpdateNickname(nickname);
    }

    public void OnColorChange(int optionValue)
    {
        Dropdown ddColor = GameObject.Find("Dropdown_Color").GetComponent<Dropdown>();
        string colorName = ddColor.options[optionValue].text;
        GameObject.FindGameObjectWithTag("Player").GetComponent<Player>().UpdateColor(colorName);
    }

}
```

29 – In the main menu, click “GameObject” => “UI” => “Input Field”. Set the name to “**InputField_Nickname**”. The control will appear on the bottom-left corner of your Game view screen.



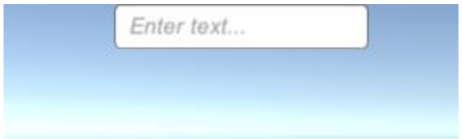
30 - In the Inspector window set both the pivot and the position to the top-center. Keep pressed Alt + Shift.


ALT + SHIFT + CLICK

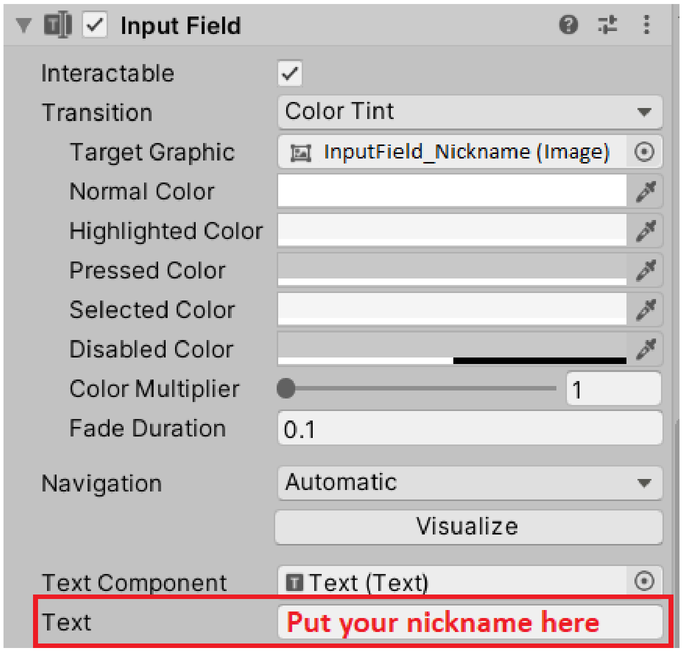
Anchor Presets

Shift: Also set pivot   Alt: Also set position

31 – The input field should be aligned like this in your Game view window:



32 – Still in the Inspector window, scroll down to the Input Field component and set the Text property value to your nickname:


Input Field

Interactable ☑

Transition: Color Tint

Target Graphic: InputField_Nickname (Image)

Normal Color

Highlighted Color

Pressed Color

Selected Color

Disabled Color

Color Multiplier: 1

Fade Duration: 0.1

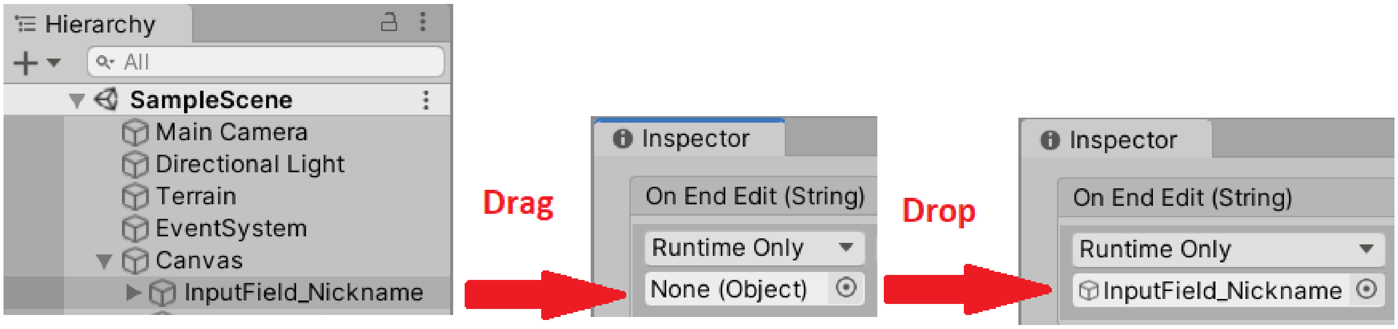
Navigation: Automatic

Visualize

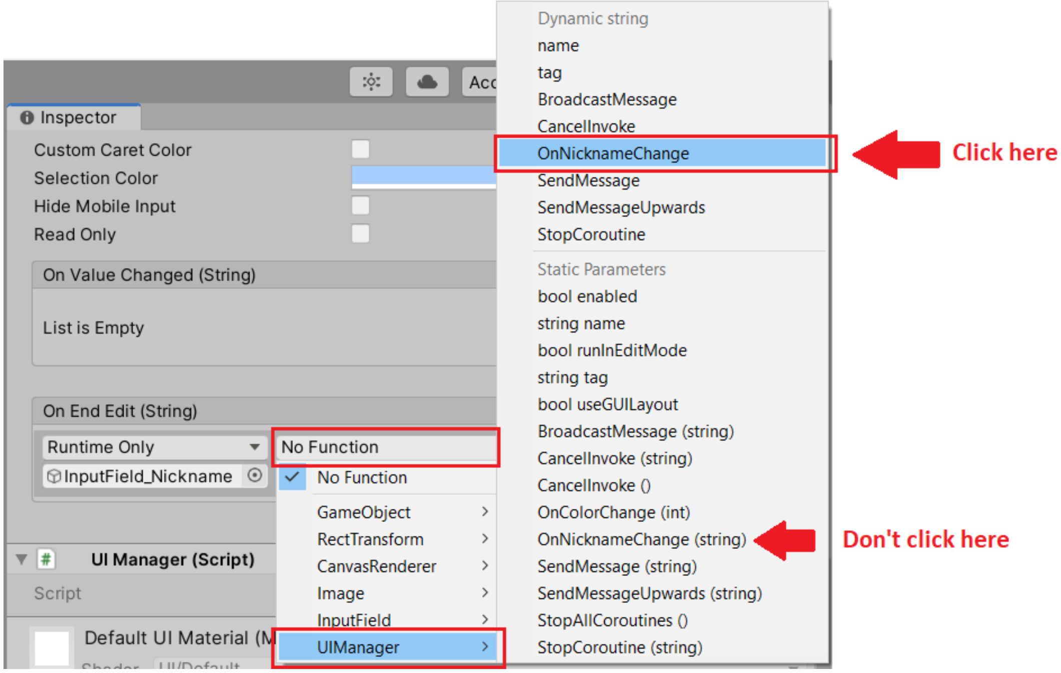
Text Component: Text (Text)

Text: Put your nickname here

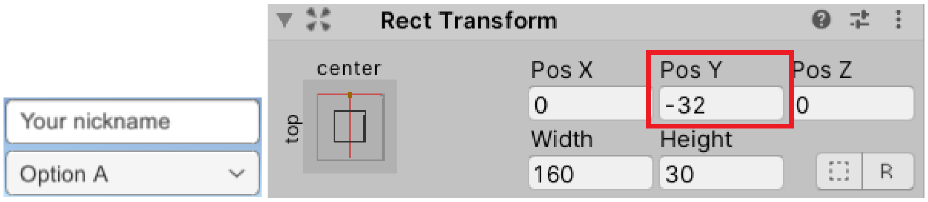
33 – Scroll down to the bottom of the Inspector window and add the “**UIManager**” component. In the “On End Edit” section click +. Drag the “**InputField_Nickname**” game object from the Hierarchy window and drop into the “**None (Object)** field:



34 – Click “No Function” => “UIManager” => “OnNicknameChage” from the Dynamic string section. Don’t click the same item from the Static Parameters section.

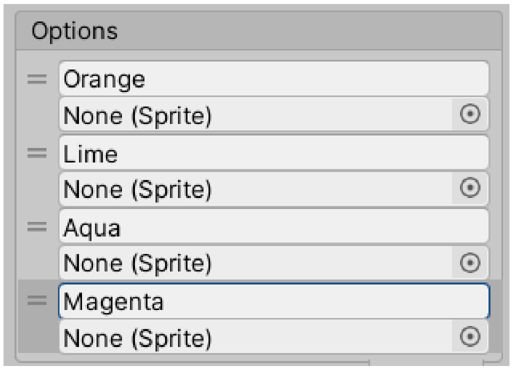


35 – In the main menu, click “GameObject” => “UI” => “Dropdown”. Set the name to “**Dropdown_Color**”. Repeat the same process as you did for the input field. Place the dropdown underneath by setting the “Pos Y” to **-32**.



Pos X	Pos Y	Pos Z
0	-32	0
Width	Height	
160	30	

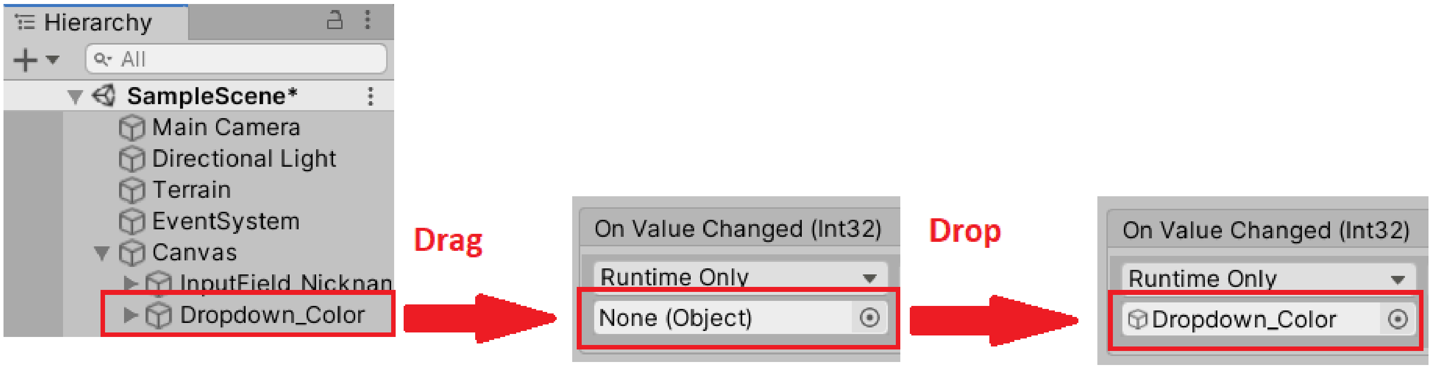
36 – Still in the Inspector window, scroll down to the Dropdown component. Remove the default options “Option A”, “Option B”, and “Option C”: use the ═ button to select the item and the − button to delete. Use the + button to add the options: “**Orange**”, “**Lime**”, “**Aqua**”, and “**Magenta**”.



Options

═ Orange
None (Sprite) ⊙
═ Lime
None (Sprite) ⊙
═ Aqua
None (Sprite) ⊙
═ Magenta
None (Sprite) ⊙

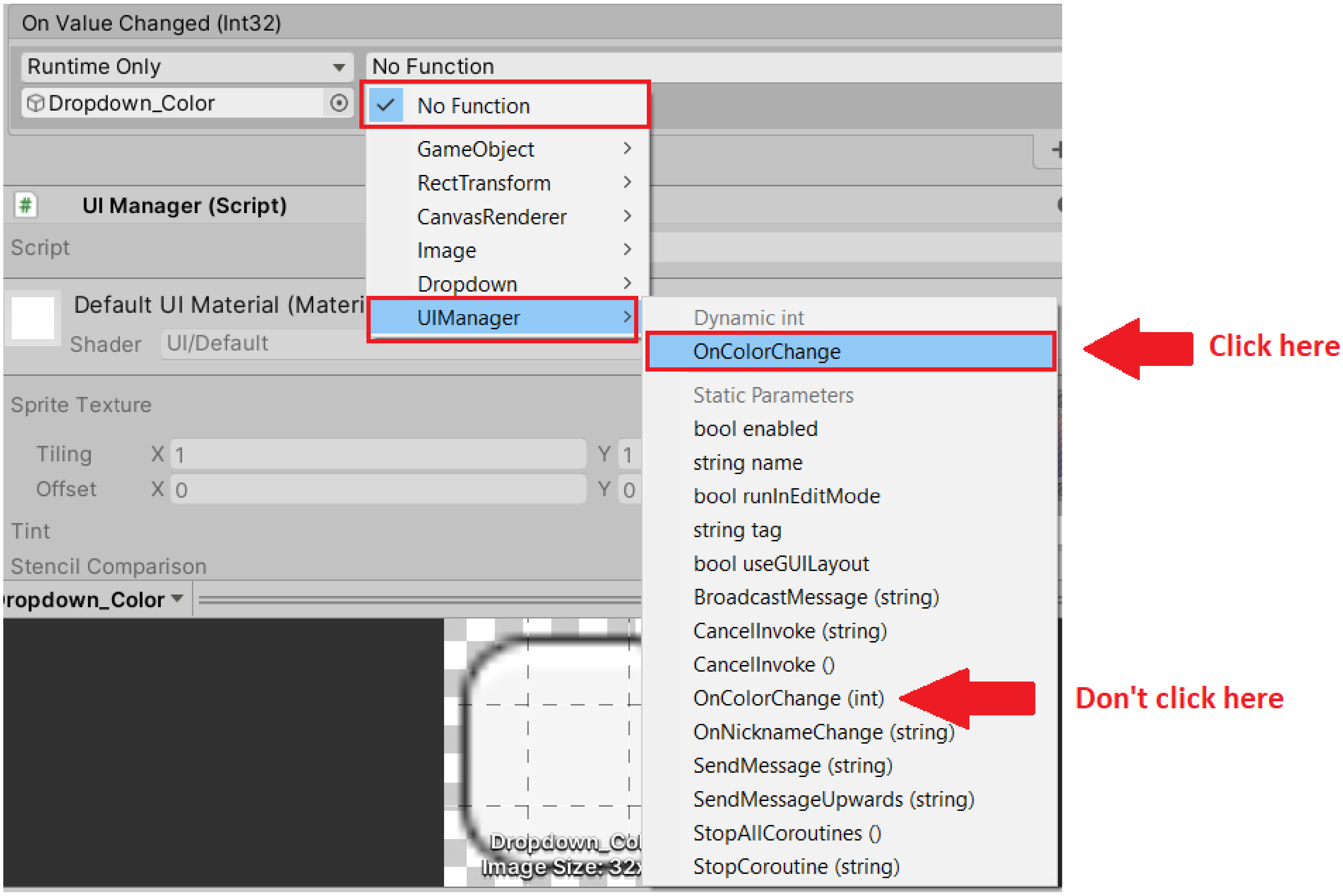
37 – Scroll down to the bottom of the Inspector window and add the “**UIManager**” component. In the “On Value Changed (Int32)” section click +. Drag the “**Dropdown_Color**” game object from the Hierarchy window and drop into the “**None (Object)** field:



On Value Changed (Int32)
Runtime Only
None (Object) ⊙

On Value Changed (Int32)
Runtime Only
Dropdown_Color ⊙

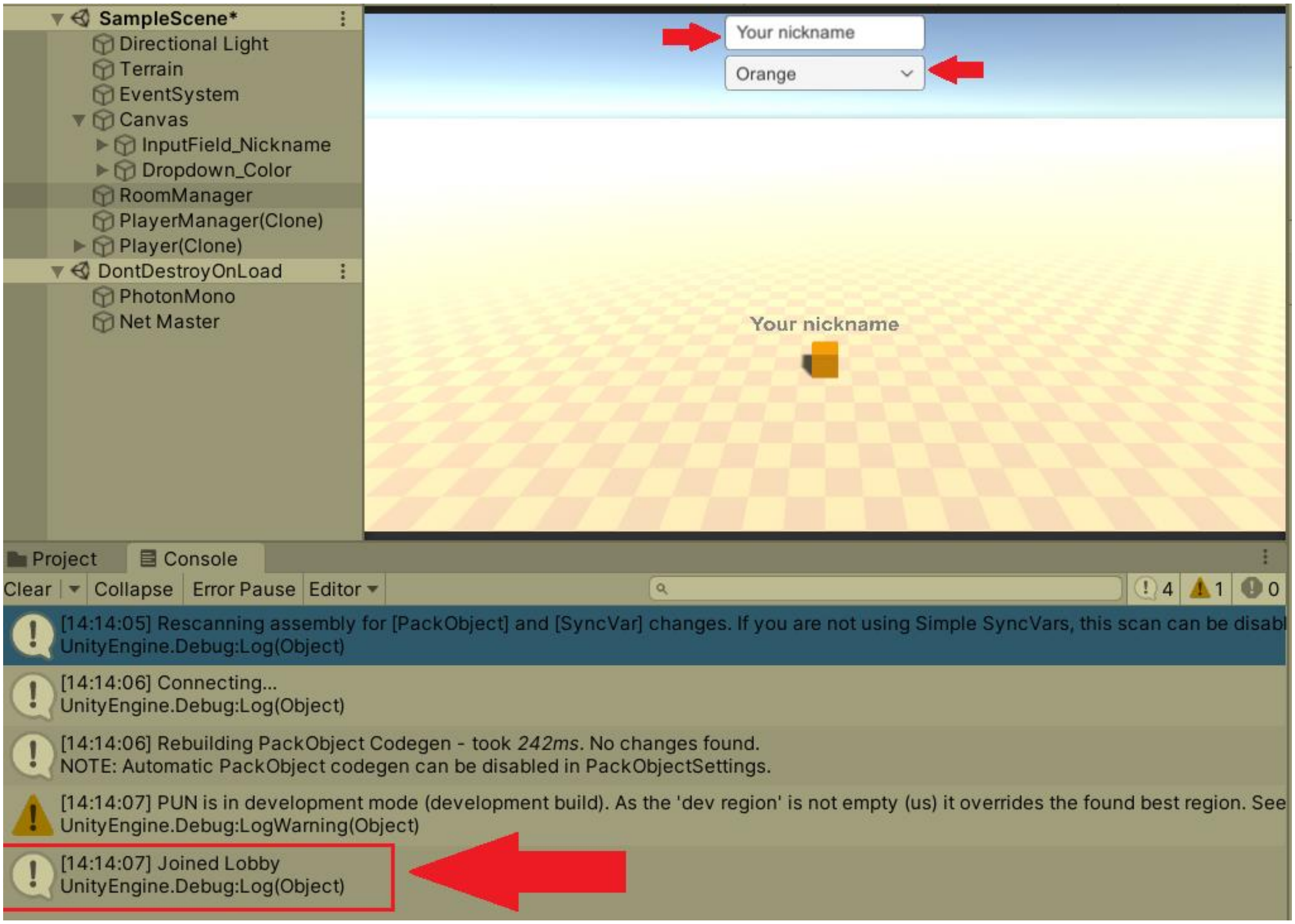
38 – Click “No Function” => “UIManager” => “OnColorChange” from the Dynamic int section. Don’t click the same item from the Static Parameters section.



Click here

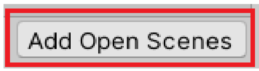
Don't click here

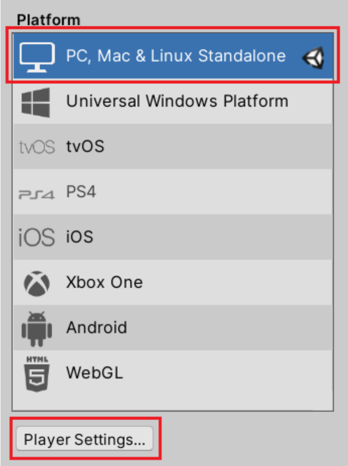
39 – Press the Play button in the Unity Editor. At first there are no players in your scene. Notice those log outputs on the Console window. After you join the lobby, your character will be spawned. Change your nickname and press enter. Also, change the color. Check if these features are working.



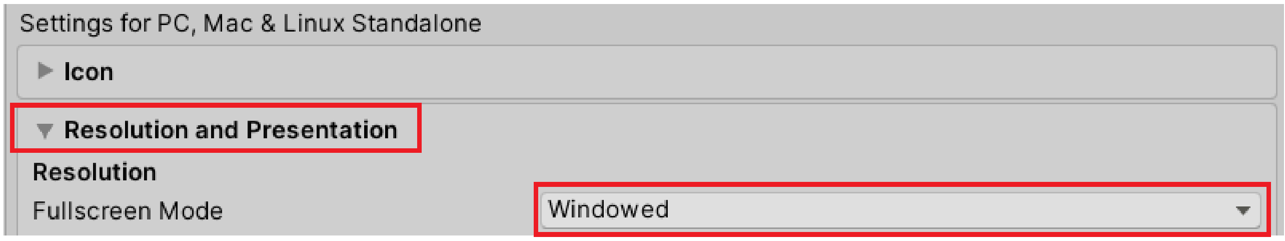
```
[14:14:05] Rescanning assembly for [PackObject] and [SyncVar] changes. If you are not using Simple SyncVars, this scan can be disabl
UnityEngine.Debug:Log(Object)
[14:14:06] Connecting...
UnityEngine.Debug:Log(Object)
[14:14:06] Rebuilding PackObject Codegen - took 242ms. No changes found.
NOTE: Automatic PackObject codegen can be disabled in PackObjectSettings.
[14:14:07] PUN is in development mode (development build). As the 'dev region' is not empty (us) it overrides the found best region. See
UnityEngine.Debug:LogWarning(Object)
[14:14:07] Joined Lobby
UnityEngine.Debug:Log(Object)
```

40 – Save your scene. In the main menu, click “File” => “Build Settings”. Click “Add open Scenes”. Make sure the current selected platform is “PC, Mac & Linux Standalone” then click “Player Settings”





41 – Expand the Resolution and Presentation section and set the Fullscreen Mode to “Windowed”. Close this window.



Settings for PC, Mac & Linux Standalone

▶ Icon

▼ Resolution and Presentation

Resolution

Fullscreen Mode: Windowed

42 – Back to the Build Settings window, click Build. Create a folder called “PCBUILD” and press “Select Folder”.

43 – Once this build process is over, open at least 2 instances of the generated executable. See how the spawned Players sync their movements, colors changes, and nickname changes.