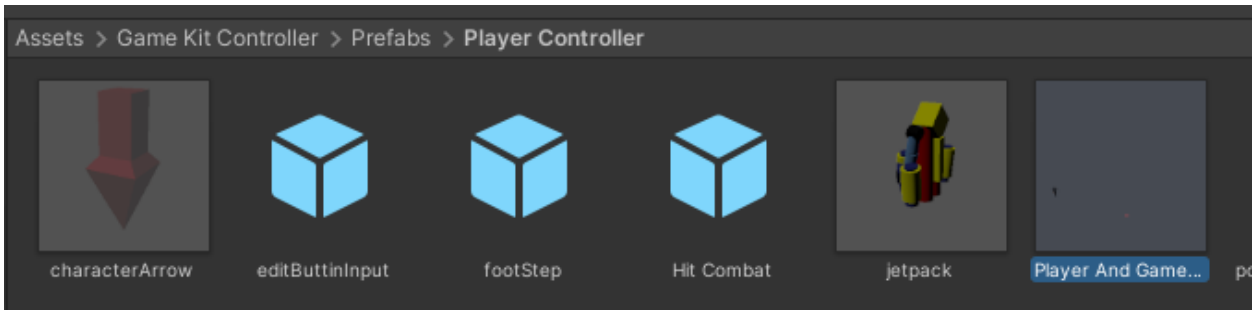
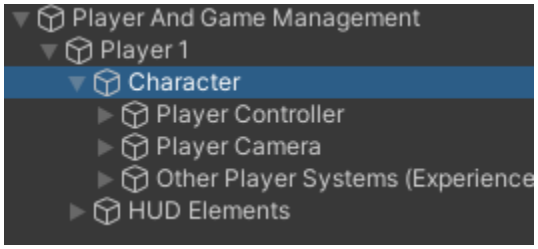
Start a new project and add GKC and Mirror.

The Mirror documentation states:
"It is important to note that Mirror does not support Network Identities on nested GameObjects. Otherwise, Mirror will emit an error.
To avoid this, ensure your parent GameObject is the only GameObject in the stack with a Network Identity. "

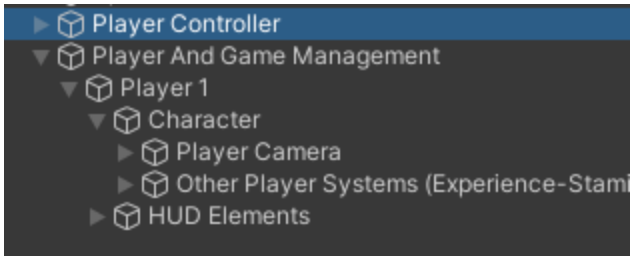
So we will create a new prefab so we can have a hierarchy that will support Mirror with a little more ease.
This is not the only change to GKC resources, so please make copies to avoid breaking changes to assets already in use.
Take the Player And Game Management prefab from this location \Assets\Game Kit Controller\Prefabs\Player Controller\Player And Game Management.prefab



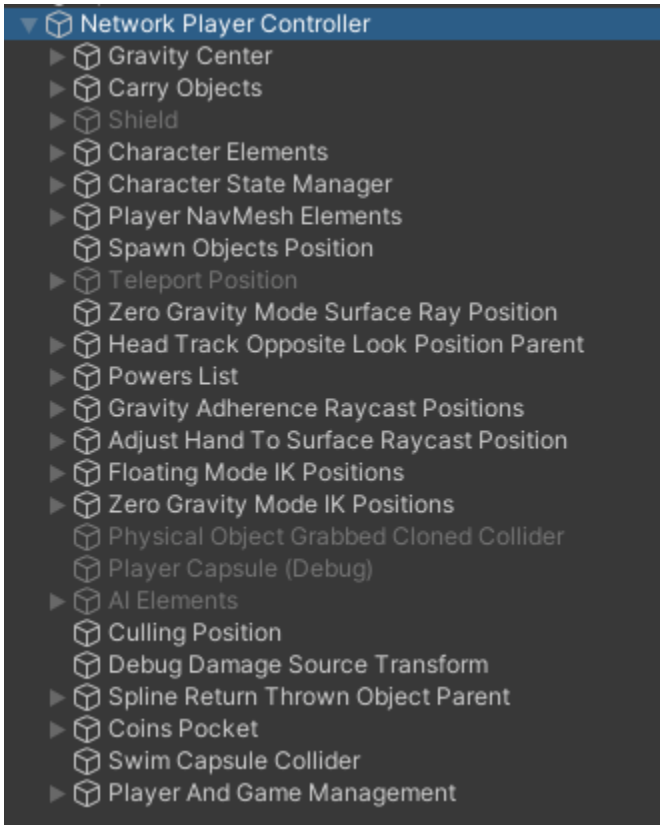
and drop it in a new scene.
Right click on it and select the "Prefab" menu > Unpack Completely.
Expand Player and Game Management
Expand Player 1
Expand Character



Drag 'Player Controller' outside the hierarchy, just above 'Player and Game Management'.

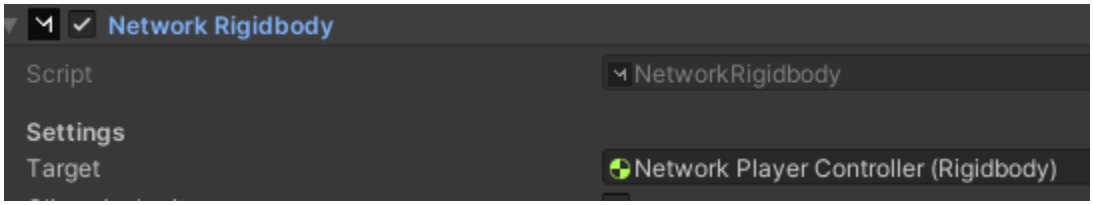


Now drag 'Player and Game Management' on top of 'Player Controller'.
Change the name of ‘Player Controller’ to “Network Player Controller”.

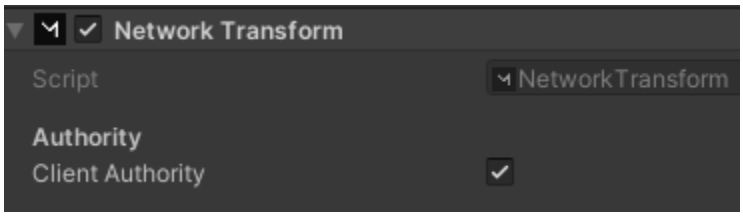


In the inspector for Network Player Controller, add the following components:

Network Identity
Network Transform
Network Animator
(depending on your movement, you may also need to add Network Rigidbody, if so, ensure the target of the Network Rigidbody is set to Network Player Controller rigidbody.)



All 3 of those Networking objects have a Client Authority boolean, make sure it is checked:



Also, add PlayerCamera from the location:
…\Assets\Mirror\Examples\MultipleAdditiveScenes\Scripts\PlayerCamera.cs
(This script is doing simple things and probably causes more trouble than it is worth. I ended up removing PlayerCamera.cs)

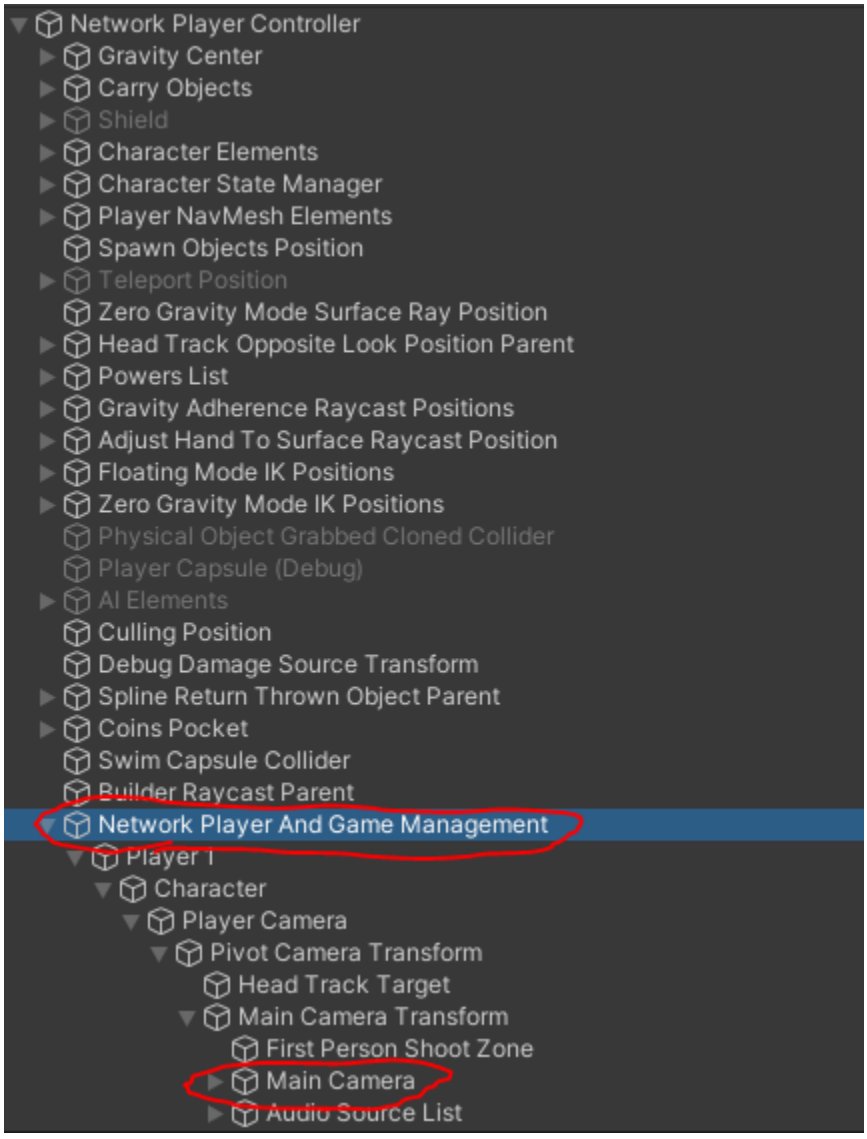
Create a new c# script and name it NetworkPlayerController.cs

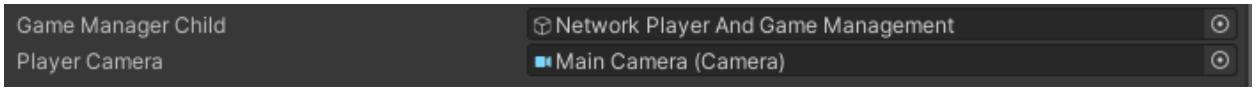
Add the following 2 variables and an Awake method to your new NetworkPlayerController.cs:

```
public GameObject GameManagerChild;
public Camera playerCamera;
private void Awake()
{
    GameManagerChild.transform.SetParent(null);
    playerCamera.enabled = false;
}
```

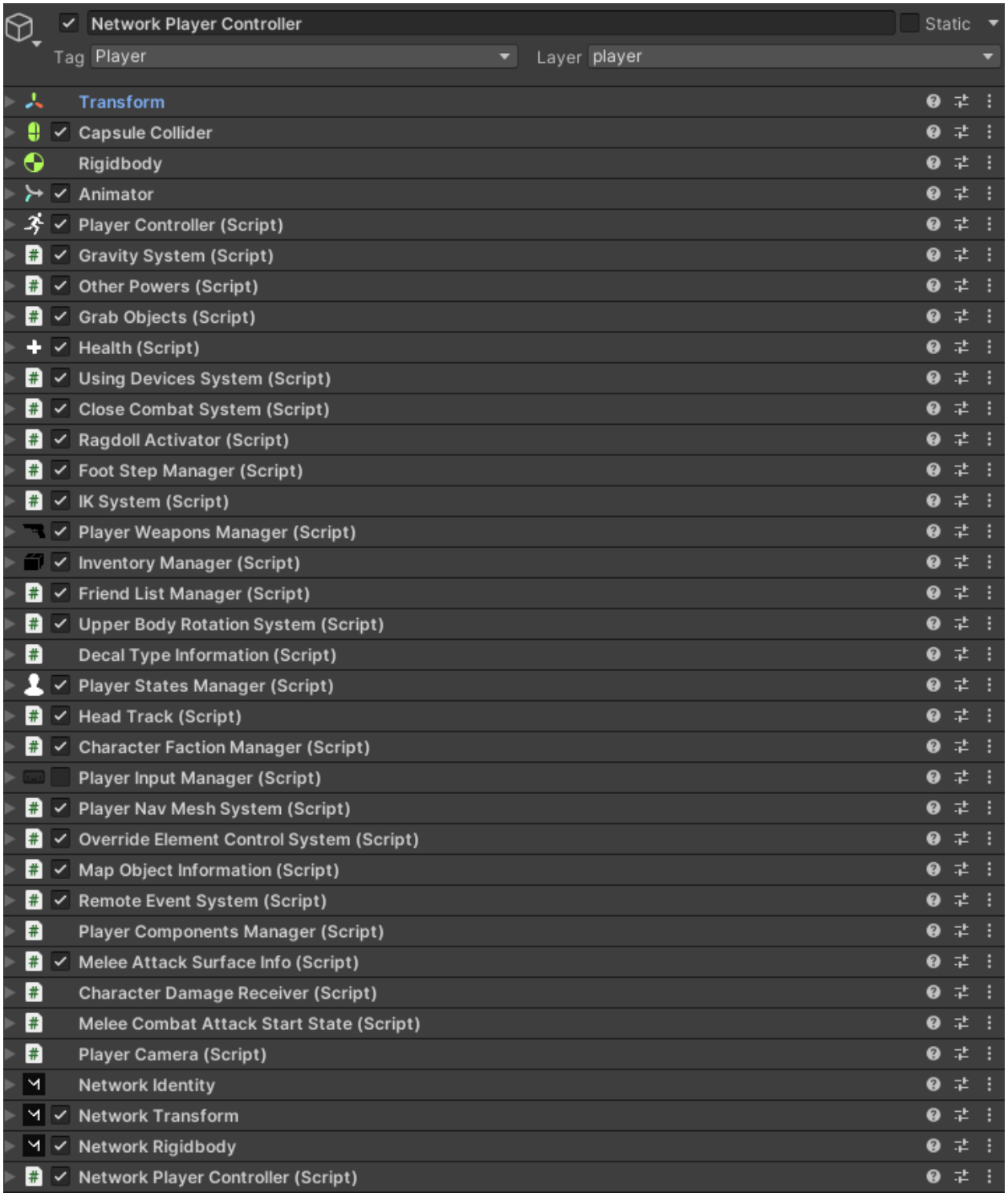
Save and Compile.

In the Inspector, assign the GameObjects to the variables we created.





This is what my Inspector looks like for my new ‘Network Player Controller’.
Save your ‘Network Player Controller’ as a Prefab.



Now open the playerController.cs script.
Add the “using Mirror;”
And change MonoBehaviour to NetworkBehavior.

```
using Mirror;

public class playerController : NetworkBehaviour
{
```

Now press CTRL + F and search for “//CALL INPUT FUNCTIONS” or canUseInput().
Add “&& isLocalPlayer” to the canUseInput() method.

```
//CALL INPUT FUNCTIONS
public bool canUseInput ()
{
    return (!jetPackEquiped && !flyModeActive && canMove && !usedByAI
        && !playerNavMeshEnabled && !driving && !swimModeActive && isLocalPlayer);
}
```

Scroll to the bottom of the playerController.cs file and add these 2 methods:

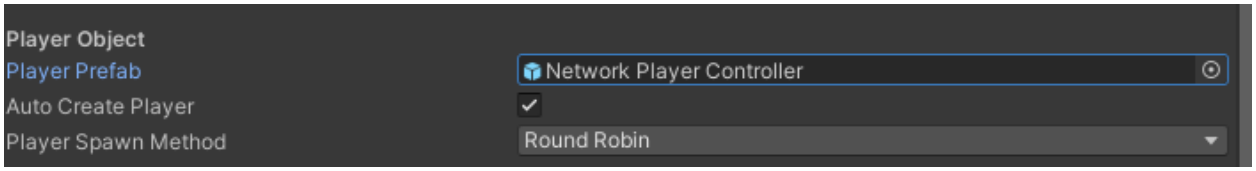
```
///Settings ported from Mirror Player Controller
///Commented code is the original, followed by additions that were needed
void OnValidate()
{
        //if (characterController == null)
        //      characterController = GetComponent<CharacterController>();
        if (playerInput == null)
        {
                playerInput = GetComponent<playerInputManager>();
        }
        //characterController.enabled = false;
        playerInput.enabled = false;
        GetComponent<Rigidbody>().isKinematic = false;
        if (GetComponent<NetworkTransform>())
        {
                GetComponent<NetworkTransform>().clientAuthority = true;
        }
}

public override void OnStartLocalPlayer()
{
        //characterController.enabled = true;

        //We have established control over the player, and can now enable things we
        //should be able to uniquely see and control
        playerInput.enabled = true;
        GetComponent<NetworkPlayerController>().playerCamera.enabled = true;
}
```

```
7971
7972    void DrawGizmos ()...
8004
8005    ///Settings ported from Mirror Player Controller
8006    void OnValidate()
8007    {
8008        //if (characterController == null)
8009        //  characterController = GetComponent<CharacterController>();
8010        if (playerInput == null)
8011        {
8012            playerInput = GetComponent<playerInputManager>();
8013        }
8014        //characterController.enabled = false;
8015        playerInput.enabled = false;
8016        GetComponent<Rigidbody>().isKinematic = false;
8017        if (GetComponent<NetworkTransform>())
8018        {
8019            GetComponent<NetworkTransform>().clientAuthority = true;
8020        }
8021    }
8022
8023    public override void OnStartLocalPlayer()
8024    {
8025        //characterController.enabled = true;
8026        playerInput.enabled = true;
8027        GetComponent<NetworkPlayerController>().playerCamera.enabled = true;
8028    }
8029}
```

Open the Mirror Example found in this location:
…\Assets\Mirror\Examples\Discovery\Scenes\Scene.unity

Select the NetworkManager from the stack. (In MultipleAdditiveScenes scene, the script is called Multi Scene Net Manager, but the process should be the same for any of the examples.)

The field “Player Prefab” has the ‘Player’ Prefab assigned to it…
Change that to our new ‘Network Player Controller’ prefab.



Follow the instructions in the README in that location, to correctly set up the scene, build, and test.