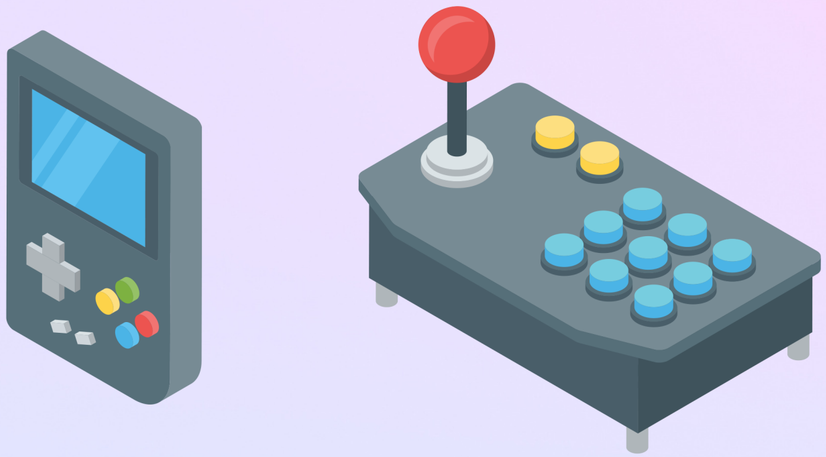


WOLVEDEN ACADEMY

NETWORKING AND MULTIPLAYER ONLINE GAMES

MULTIPLAYER NEW ERA

BY PONGSATHORN KIATTICHAROENPORN (MEW)

WEEK 8 : CONNECTING ONLINE 3

Connecting Online 3

Topic in this week

Workshop

- Player Name Selection
- Connection Approval
- Handling Connections
- Networking Improvements
- Shutting Down Cleanly

Player Name Selection

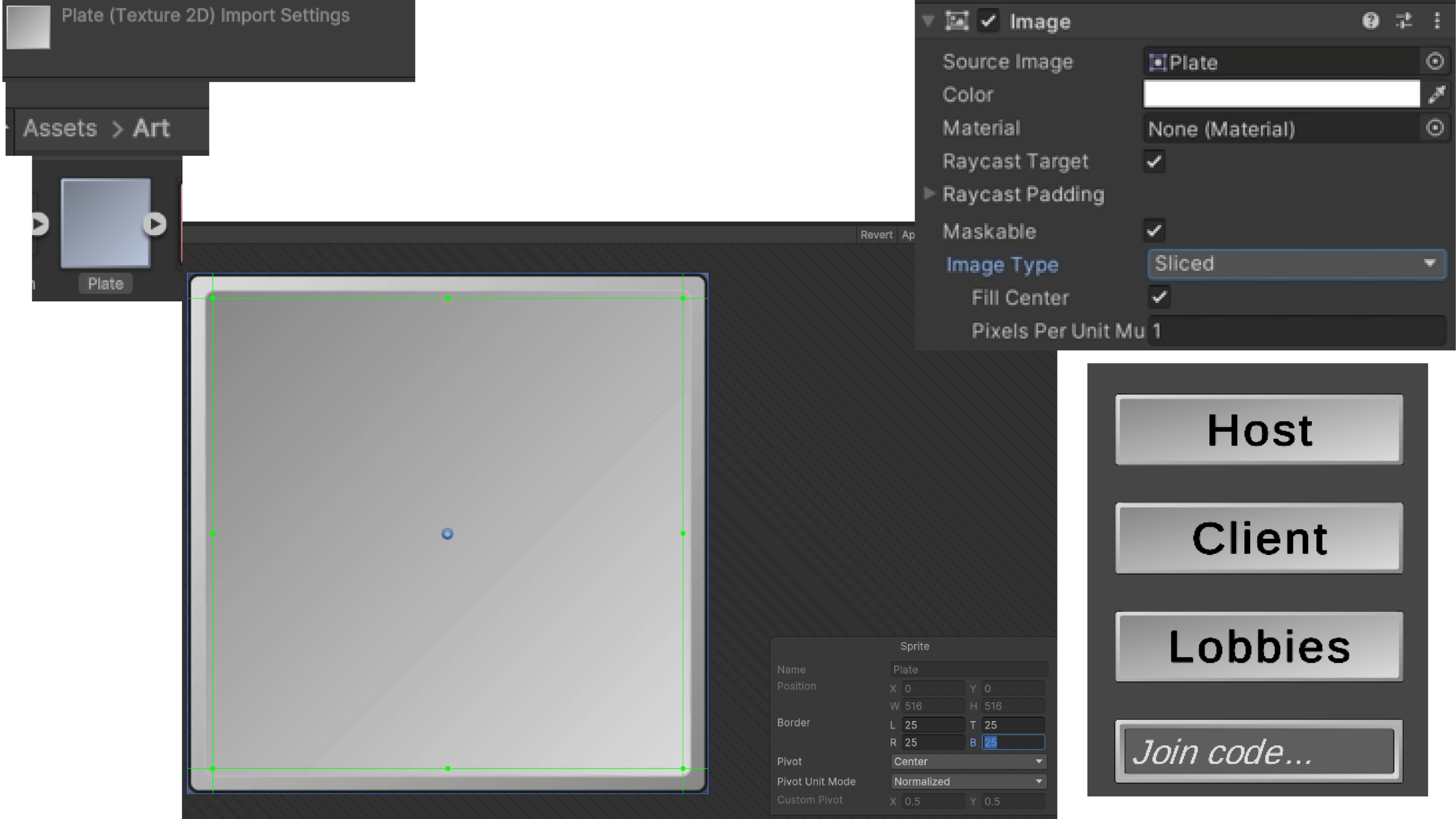
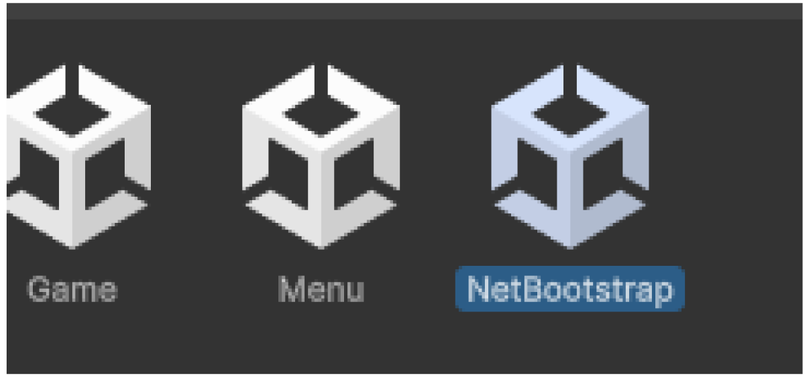
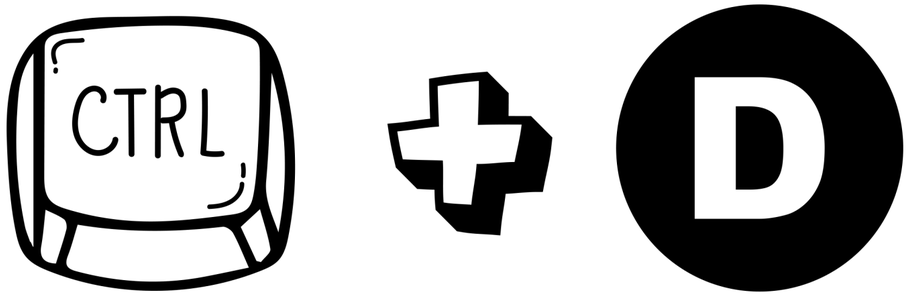
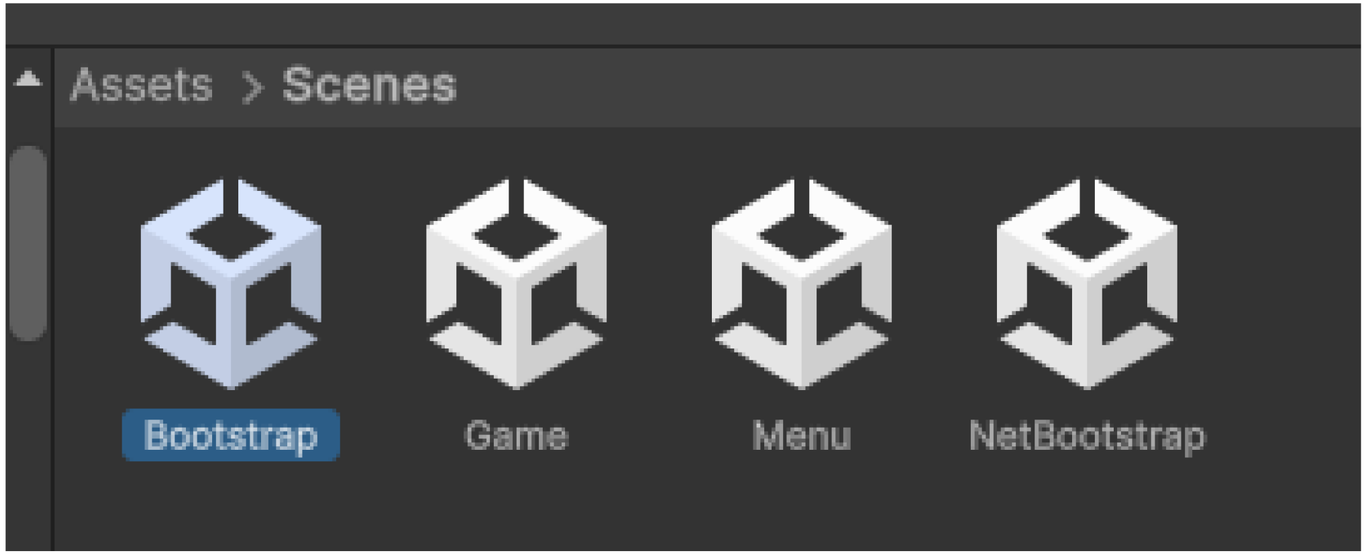


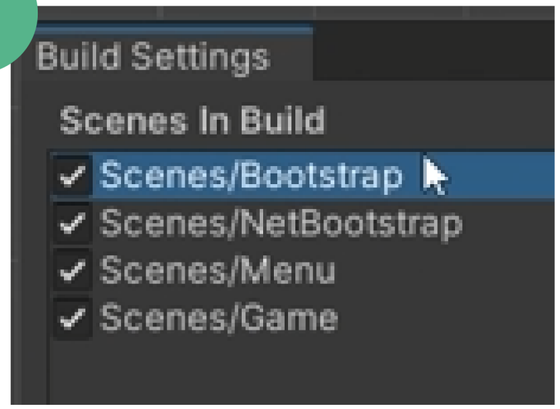
Plate (Texture 2D) Import Settings
Assets > Art
Plate
Revert
Ap
Image
Source Image
Plate
Color
Material
None (Material)
Raycast Target
Raycast Padding
Maskable
Image Type
Sliced
Fill Center
Pixels Per Unit Mu
1
Sprite
Name
Plate
Position
X 0
Y 0
W 516
H 516
Border
L 25
T 25
R 25
B 25
Pivot
Center
Pivot Unit Mode
Normalized
Custom Pivot
X 0.5
Y 0.5
Host
Client
Lobbies
Join code...


Game
Menu
NetBootstrap

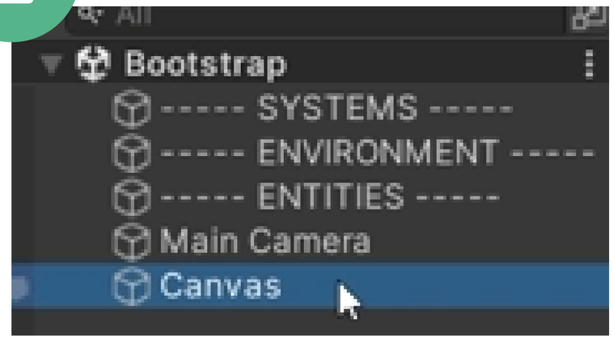

CTRL
D


Assets > Scenes
Bootstrap
Game
Menu
NetBootstrap

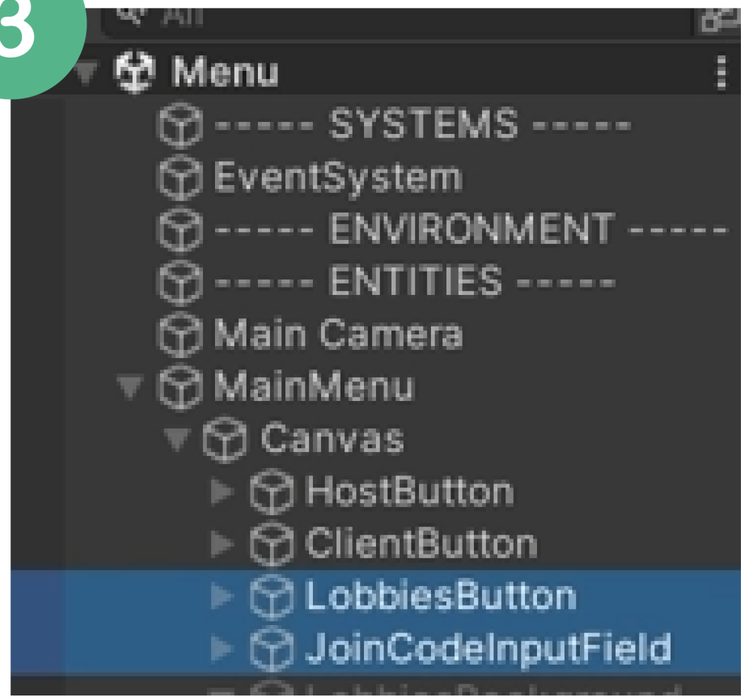
1


Build Settings
Scenes In Build
Scenes/Bootstrap
Scenes/NetBootstrap
Scenes/Menu
Scenes/Game

2


Bootstrap
----- SYSTEMS -----
----- ENVIRONMENT -----
----- ENTITIES -----
Main Camera
Canvas

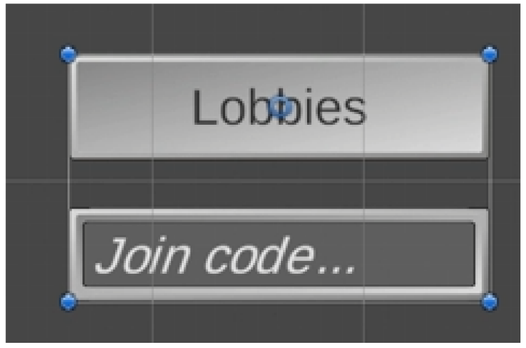
3


Menu
----- SYSTEMS -----
EventSystem
----- ENVIRONMENT -----
----- ENTITIES -----
Main Camera
MainMenu
Canvas
HostButton
ClientButton
LobbiesButton
JoinCodeInputField

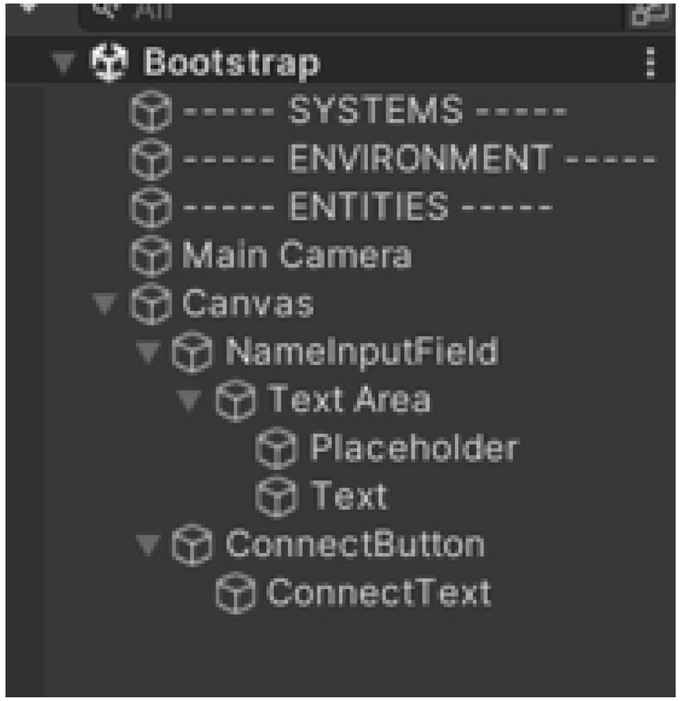

Ctrl
+
C

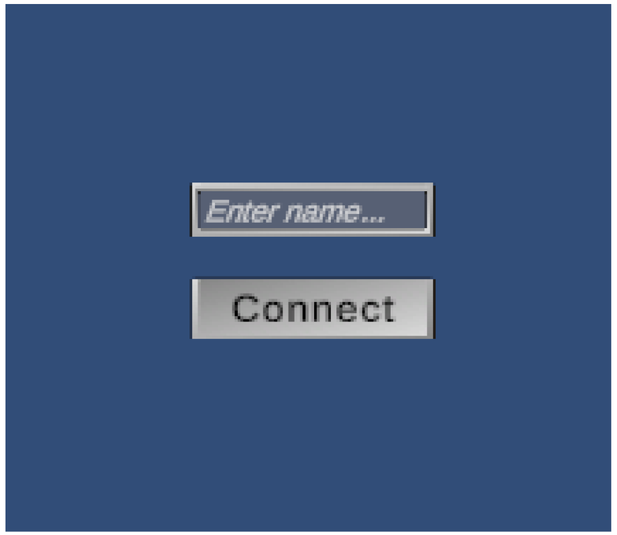
4

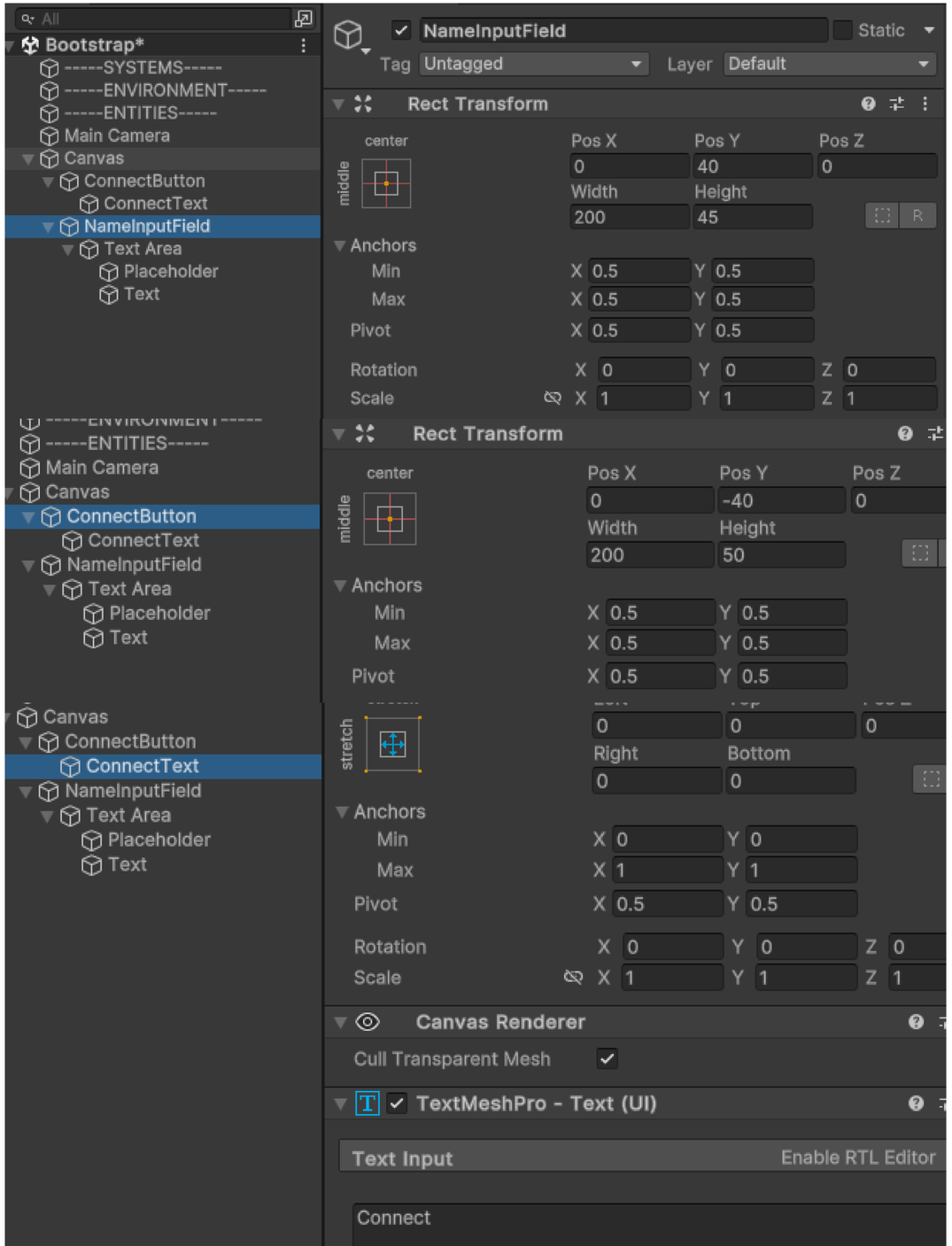

----- SYSTEMS -----
----- ENVIRONMENT -----
----- ENTITIES -----
Main Camera
Canvas
JoinCodeInputField
LobbiesButton

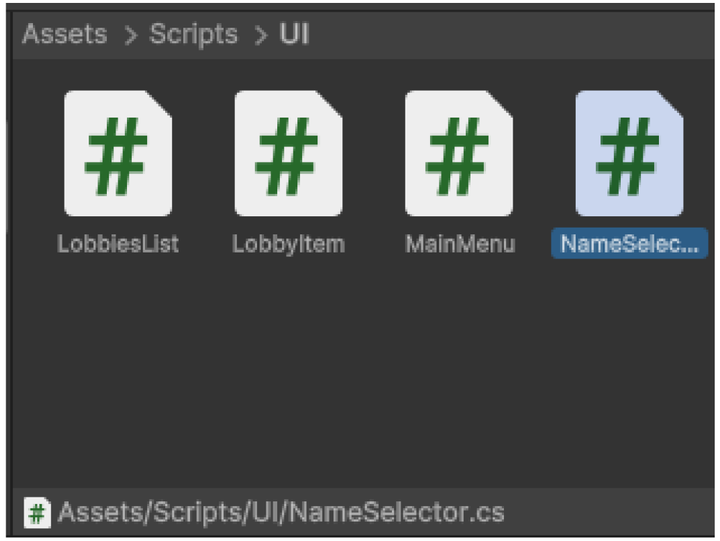

Lobbies
Join code...


Ctrl
+
V


Bootstrap
----- SYSTEMS -----
----- ENVIRONMENT -----
----- ENTITIES -----
Main Camera
Canvas
NameInputField
Text Area
Placeholder
Text
ConnectButton
ConnectText


Enter name...
Connect


Bootstrap*
-----SYSTEMS-----
-----ENVIRONMENT-----
-----ENTITIES-----
Main Camera
Canvas
ConnectButton
ConnectText
NameInputField
Text Area
Placeholder
Text
NameInputField
Static
Tag Untagged
Layer Default
Rect Transform
center
middle
Pos X 0
Pos Y 40
Pos Z 0
Width 200
Height 45
Anchors
Min X 0.5 Y 0.5
Max X 0.5 Y 0.5
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Rect Transform
center
middle
Pos X 0
Pos Y -40
Pos Z 0
Width 200
Height 50
Anchors
Min X 0.5 Y 0.5
Max X 0.5 Y 0.5
Pivot X 0.5 Y 0.5
stretch
Left 0
Top 0
Pos Z 0
Right 0
Bottom 0
Anchors
Min X 0 Y 0
Max X 1 Y 1
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Canvas Renderer
Cull Transparent Mesh
TextMeshPro - Text (UI)
Text Input
Enable RTL Editor
Connect



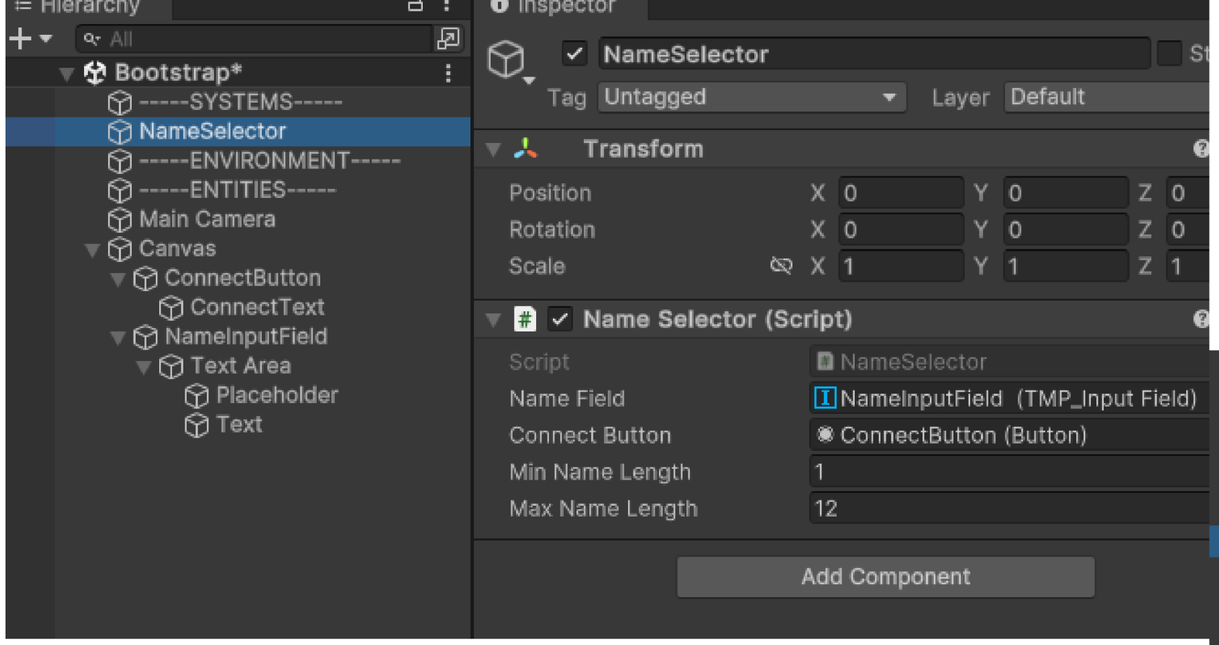

```
using System.Collections;
using System.Collections.Generic;
using TMPro;
using UnityEngine;
using UnityEngine.UI;

Unity Script | 0 references
public class NameSelector : MonoBehaviour
{
    [SerializeField] private TMP_InputField nameField;
    [SerializeField] private Button connectButton;
    [SerializeField] private int minNameLength = 1;
    [SerializeField] private int maxNameLength = 12;

```

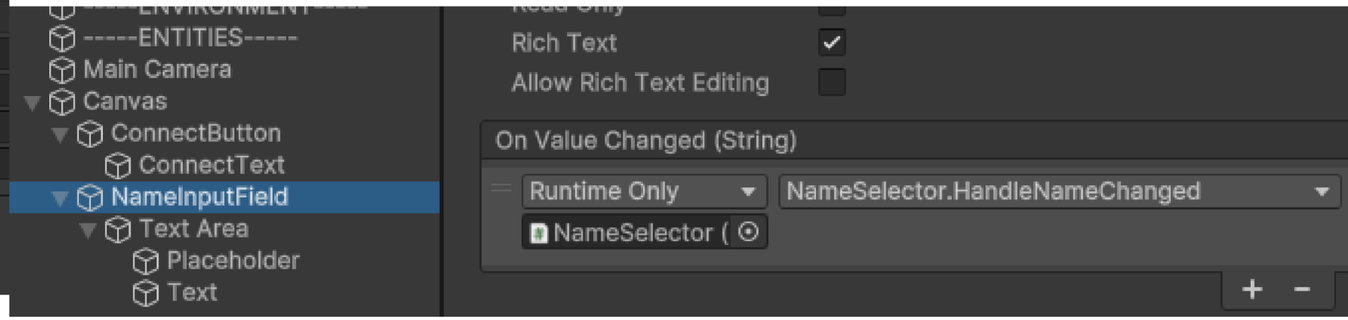
```
private const string PlayerNameKey = "PlayerName";
Unity Message | 0 references
private void Start()
{
    if(SystemInfo.graphicsDeviceType == UnityEngine.Rendering.GraphicsDeviceType.Null)
    {
        return;
    }
    nameField.text = PlayerPrefs.GetString(PlayerNameKey, string.Empty);
    HandleNameChanged();
}

1 reference
public void HandleNameChanged()
{
    connectButton.interactable =
        nameField.text.Length >= minNameLength &&
        nameField.text.Length <= maxNameLength;
}
```

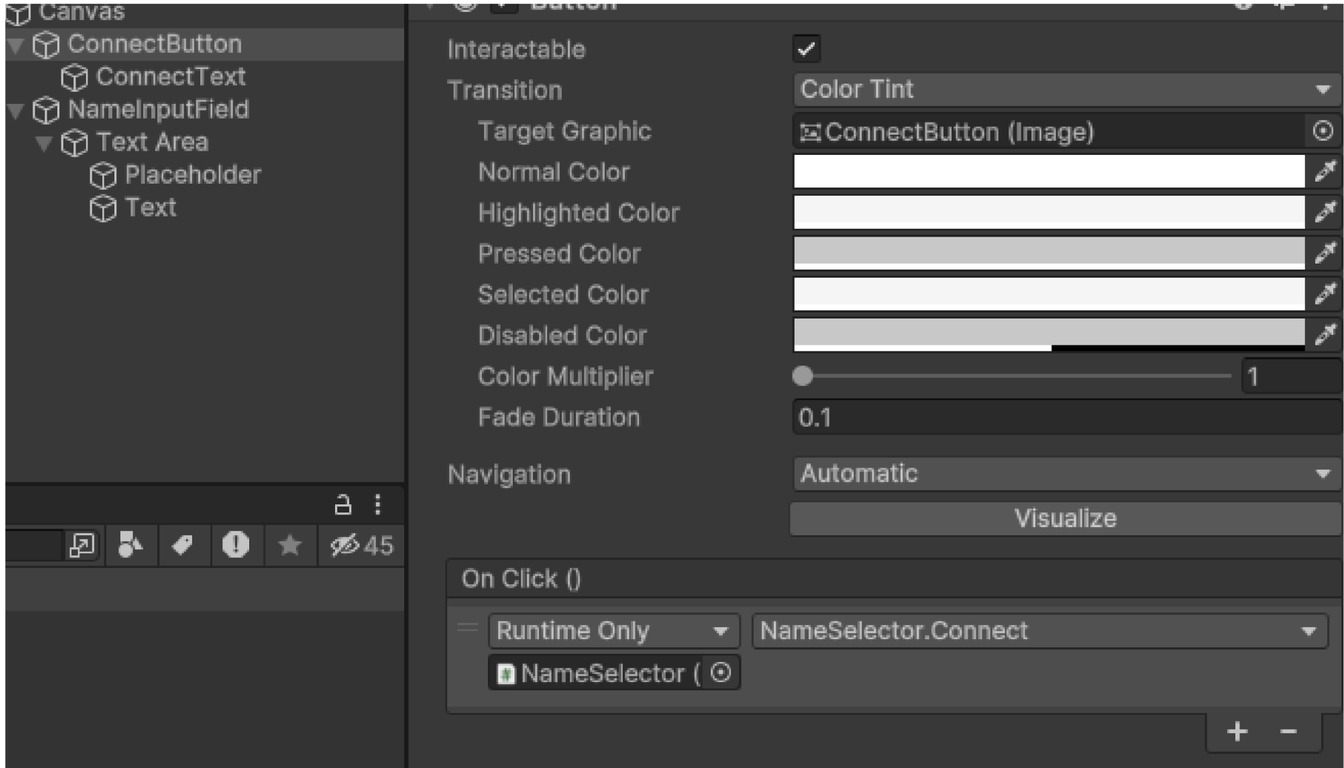
Save The Name

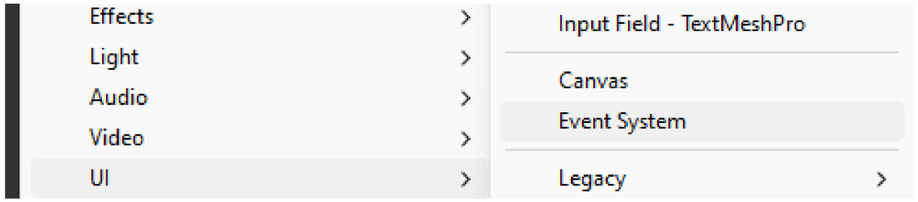
- Save the name with **PlayerPrefs.SetString**
- Load the next scene
- In **Start()**, also load the next scene if we are a headless server

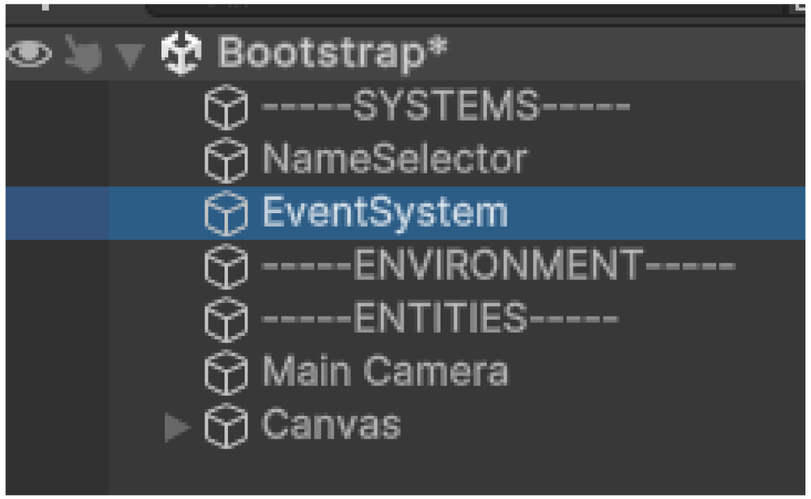
```
using System.Collections;
using System.Collections.Generic;
using TMPro;
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

Unity Script | 0 references
public class NameSelector : MonoBehaviour
{
    [SerializeField] private TMP_InputField nameField;
    [SerializeField] private Button connectButton;
    [SerializeField] private int minNameLength = 1;
    [SerializeField] private int maxNameLength = 12;

    private const string PlayerNameKey = "PlayerName";

    Unity Message | 0 references
    private void Start()
    {
        if(SystemInfo.graphicsDeviceType == UnityEngine.Rendering.GraphicsDeviceType.Null)
        {
            SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
            return;
        }
        nameField.text = PlayerPrefs.GetString(PlayerNameKey, string.Empty);
        HandleNameChanged();
    }

    1 reference
    public void HandleNameChanged()
    {
        connectButton.interactable =
            nameField.text.Length >= minNameLength &&
            nameField.text.Length <= maxNameLength;
    }

    0 references
    public void Connect()
    {
        PlayerPrefs.SetString(PlayerNameKey, nameField.text);
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
    }
}
```

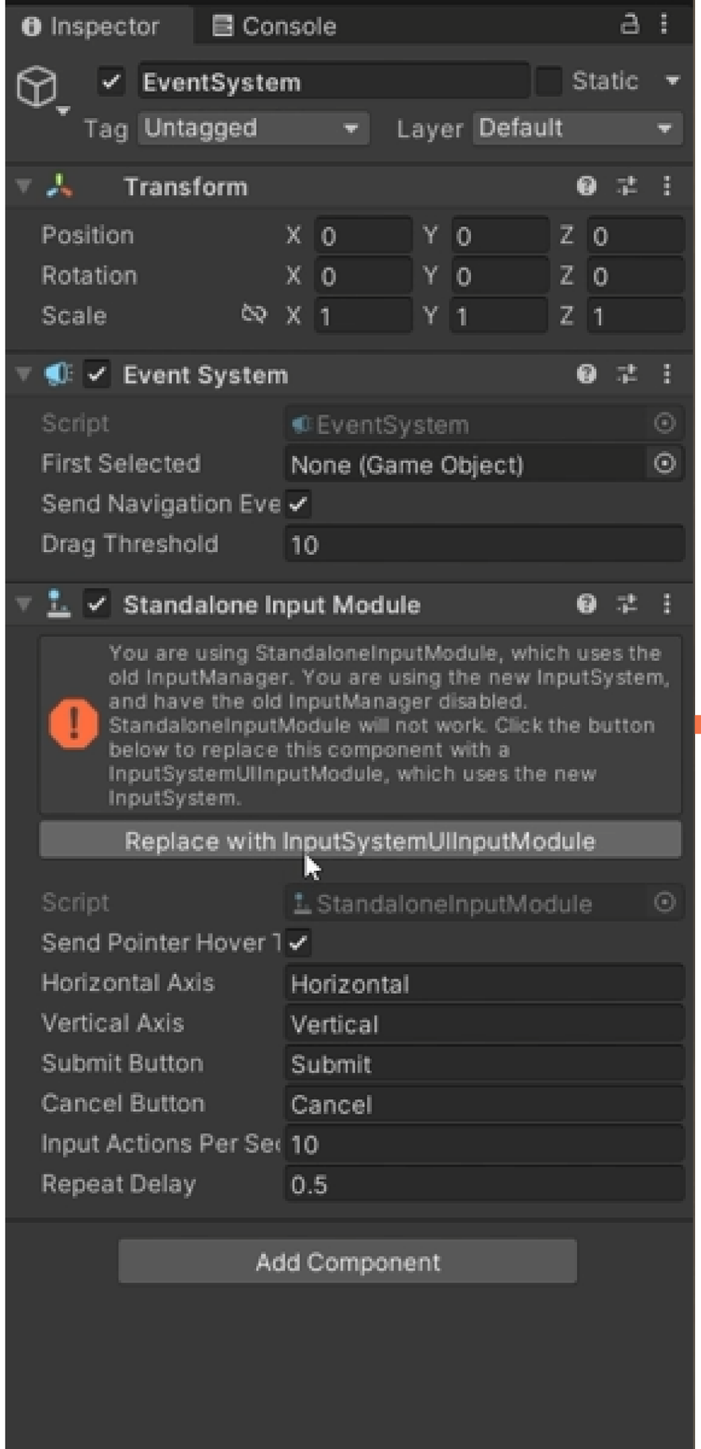

Hierarchy
All
Bootstrap*
-----SYSTEMS-----
NameSelector
-----ENVIRONMENT-----
-----ENTITIES-----
Main Camera
Canvas
ConnectButton
ConnectText
NameInputField
Text Area
Placeholder
Text
Inspector
NameSelector
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Name Selector (Script)
Script NameSelector
Name Field NameInputField (TMP_Input Field)
Connect Button ConnectButton (Button)
Min Name Length 1
Max Name Length 12
Add Component

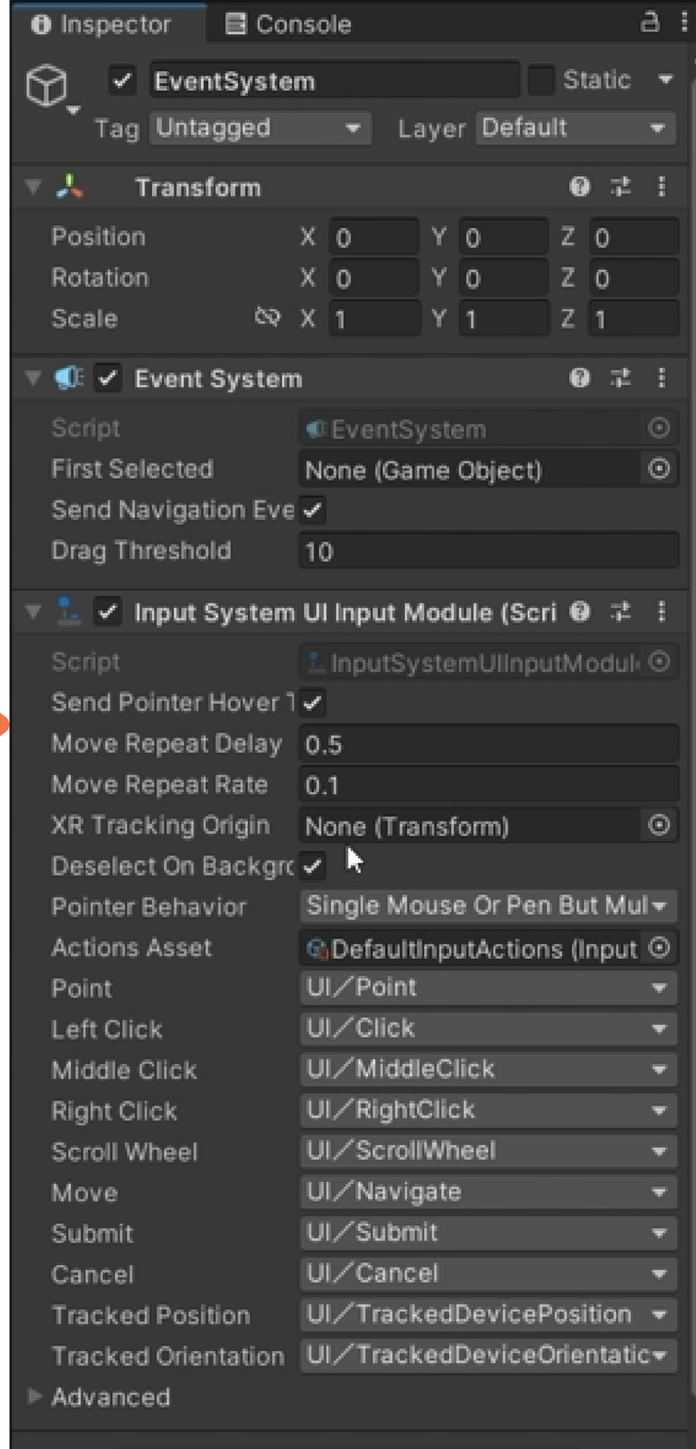

-----ENTITIES-----
Main Camera
Canvas
ConnectButton
ConnectText
NameInputField
Text Area
Placeholder
Text
Rich Text
Allow Rich Text Editing
On Value Changed (String)
Runtime Only
NameSelector.HandleNameChanged
NameSelector (


Canvas
ConnectButton
ConnectText
NameInputField
Text Area
Placeholder
Text
45
Interactable
Transition
Color Tint
Target Graphic
ConnectButton (Image)
Normal Color
Highlighted Color
Pressed Color
Selected Color
Disabled Color
Color Multiplier
1
Fade Duration
0.1
Navigation
Automatic
Visualize
On Click ()
Runtime Only
NameSelector.Connect
NameSelector (

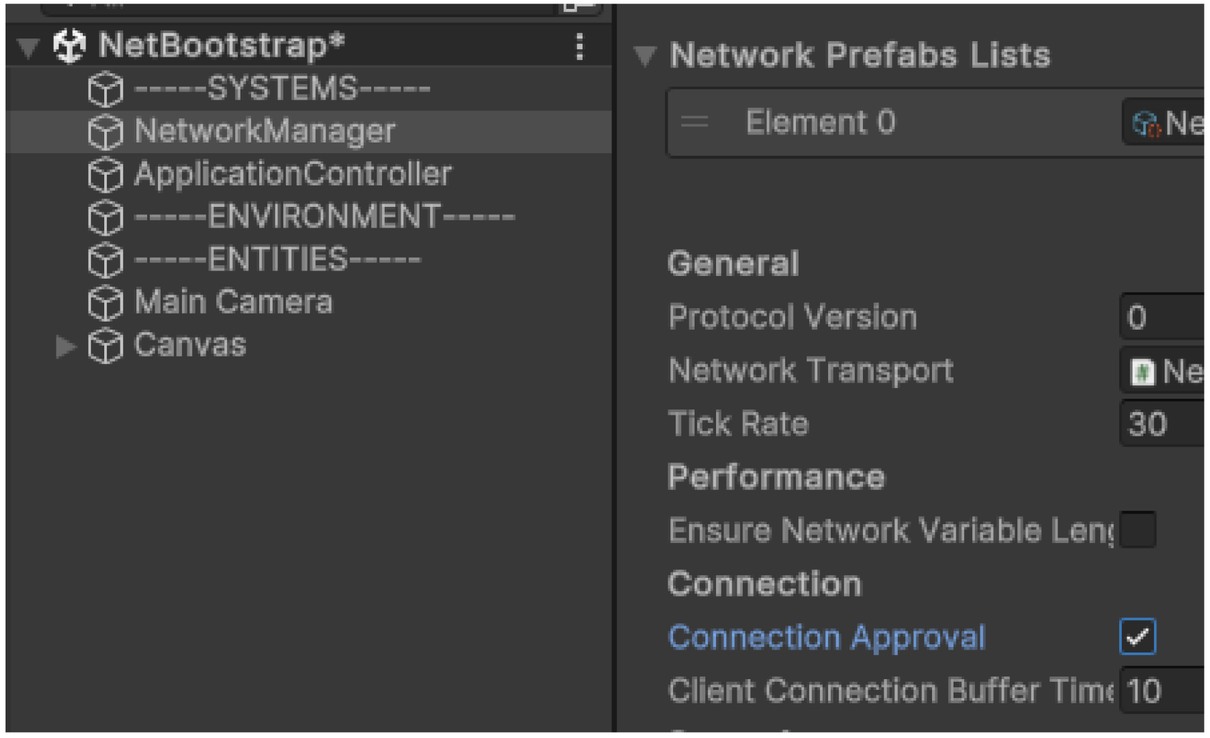

Effects
Light
Audio
Video
UI
Input Field - TextMeshPro
Canvas
Event System
Legacy

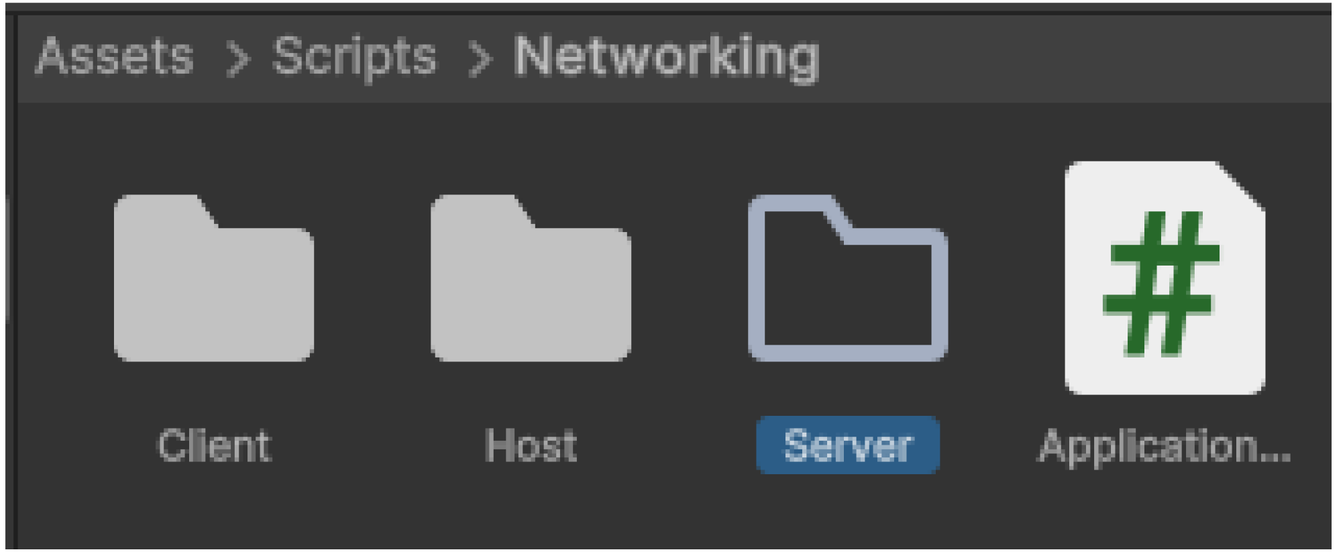

Bootstrap*
-----SYSTEMS-----
NameSelector
EventSystem
-----ENVIRONMENT-----
-----ENTITIES-----
Main Camera
Canvas

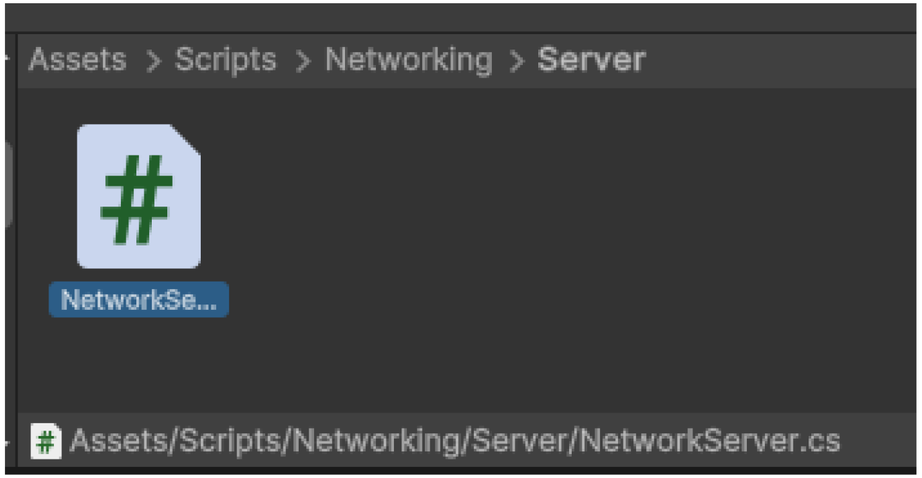

Inspector
Console
EventSystem
Static
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Event System
Script EventSystem
First Selected None (Game Object)
Send Navigation Eve
Drag Threshold 10
Standalone Input Module
You are using StandaloneInputModule, which uses the old InputManager. You are using the new InputSystem, and have the old InputManager disabled. StandaloneInputModule will not work. Click the button below to replace this component with a InputSystemUIInputModule, which uses the new InputSystem.
Replace with InputSystemUIInputModule
Script StandaloneInputModule
Send Pointer Hover T
Horizontal Axis Horizontal
Vertical Axis Vertical
Submit Button Submit
Cancel Button Cancel
Input Actions Per Se 10
Repeat Delay 0.5
Add Component


Inspector
Console
EventSystem
Static
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Event System
Script EventSystem
First Selected None (Game Object)
Send Navigation Eve
Drag Threshold 10
Input System UI Input Module (Scri
Script InputSystemUIInputModul
Send Pointer Hover T
Move Repeat Delay 0.5
Move Repeat Rate 0.1
XR Tracking Origin None (Transform)
Deselect On Backgrc
Pointer Behavior Single Mouse Or Pen But Mul
Actions Asset DefaultInputActions (Input
Point UI/Point
Left Click UI/Click
Middle Click UI/MiddleClick
Right Click UI/RightClick
Scroll Wheel UI/ScrollWheel
Move UI/Navigate
Submit UI/Submit
Cancel UI/Cancel
Tracked Position UI/TrackedDevicePosition
Tracked Orientation UI/TrackedDeviceOrientatic
Advanced

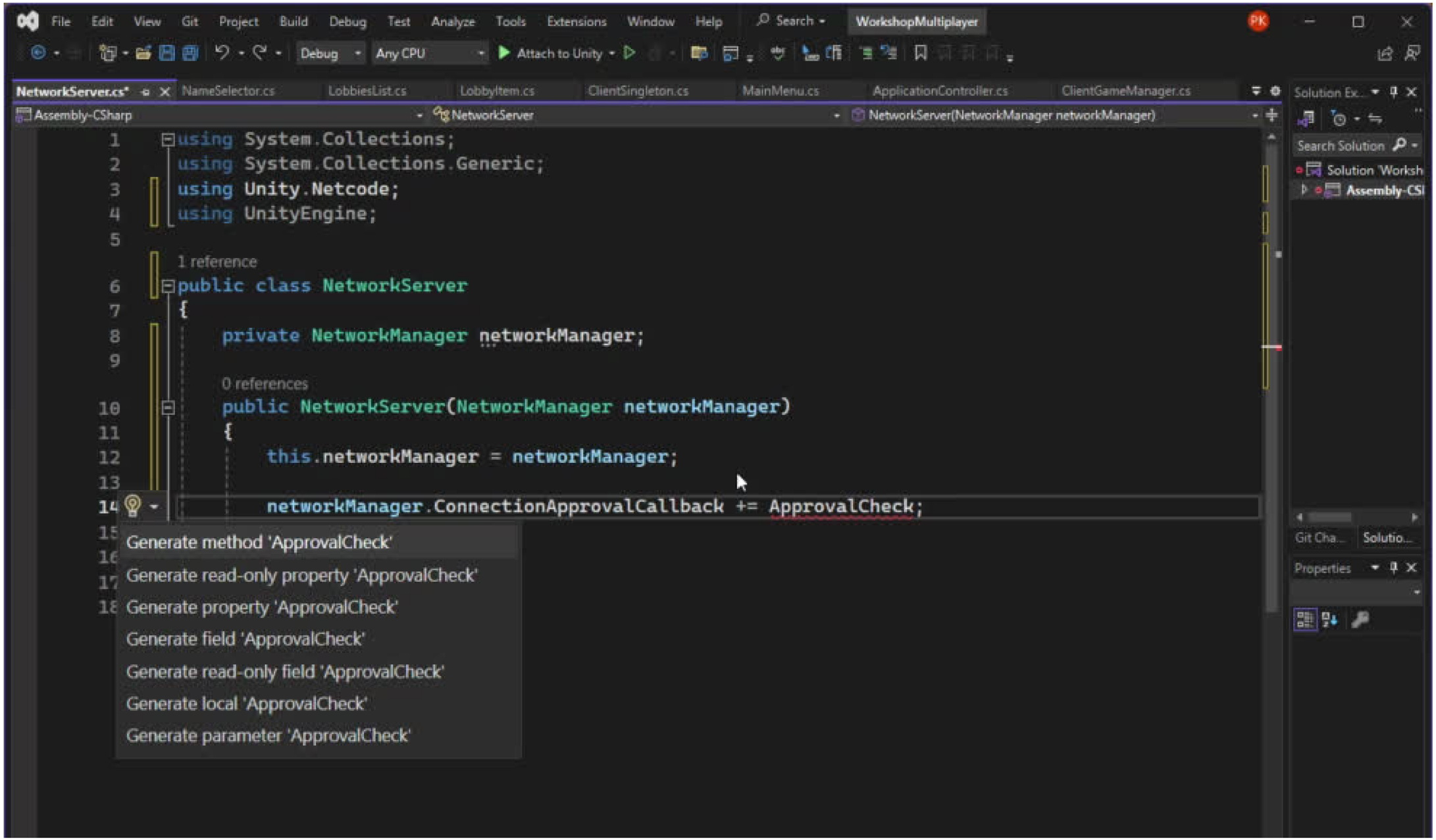
Connection Approval

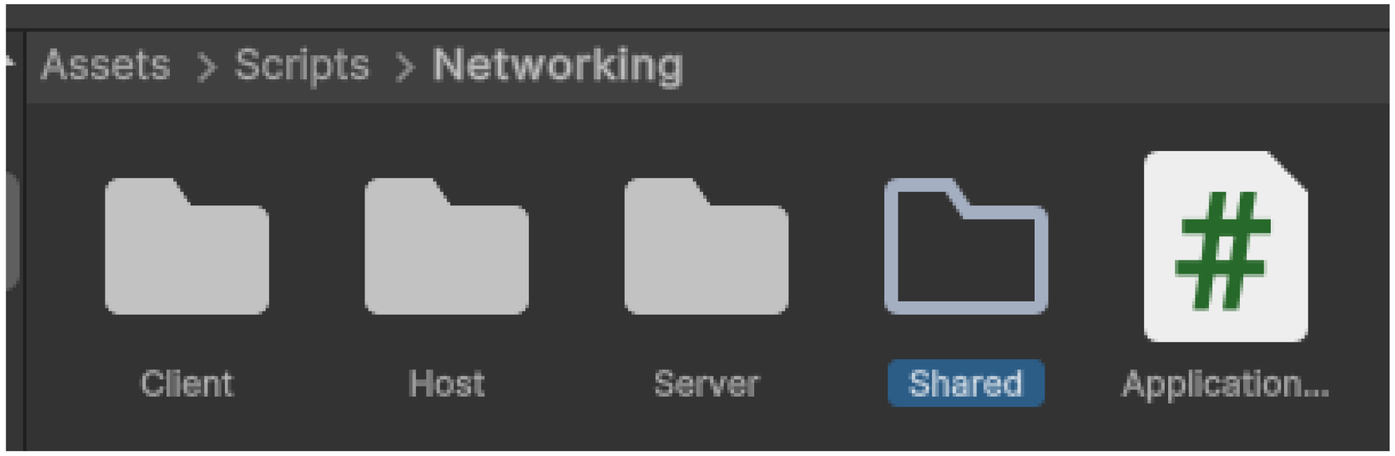

NetBootstrap*
-----SYSTEMS-----
NetworkManager
ApplicationController
-----ENVIRONMENT-----
-----ENTITIES-----
Main Camera
Canvas
Network Prefabs Lists
Element 0
General
Protocol Version
0
Network Transport
Tick Rate
30
Performance
Ensure Network Variable Len
Connection
Connection Approval
Client Connection Buffer Time
10

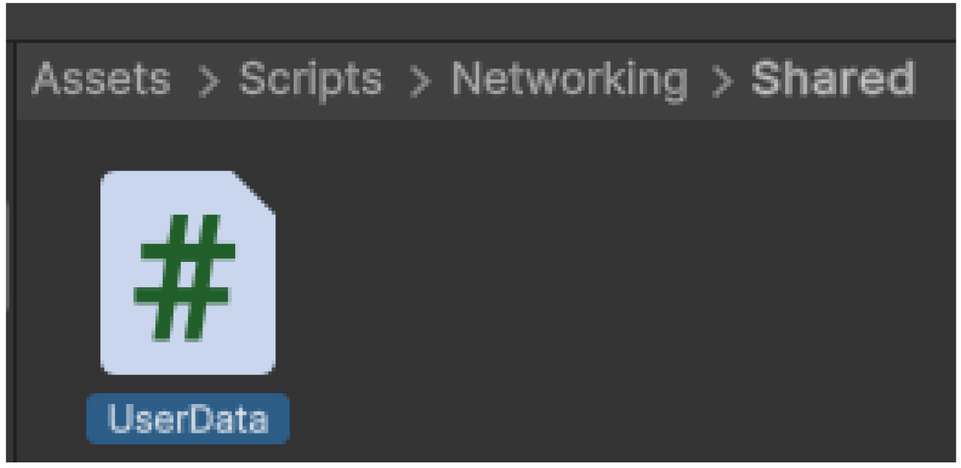

Assets > Scripts > Networking
Client
Host
Server
Application...




```
1   using System.Collections;
2   using System.Collections.Generic;
3   using Unity.Netcode;
4   using UnityEngine;
5
        1 reference
6   public class NetworkServer
7   {
8       private NetworkManager networkManager;
9
            0 references
10      public NetworkServer(NetworkManager networkManager)
11      {
12          this.networkManager = networkManager;
13
14          networkManager.ConnectionApprovalCallback += ApprovalCheck;
15      }
16
17  }
18
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
WorkshopMultiplayer
Debug
Any CPU
Attach to Unity
NetworkServer.cs*
NameSelector.cs
LobbiesList.cs
LobbyItem.cs
ClientSingleton.cs
MainMenu.cs
ApplicationController.cs
ClientGameManager.cs
Assembly-CSharp
NetworkServer
NetworkServer(NetworkManager networkManager)
using System.Collections;
using System.Collections.Generic;
using Unity.Netcode;
using UnityEngine;
1 reference
public class NetworkServer
{
private NetworkManager networkManager;
0 references
public NetworkServer(NetworkManager networkManager)
{
this.networkManager = networkManager;
networkManager.ConnectionApprovalCallback += ApprovalCheck;
Generate method 'ApprovalCheck'
Generate read-only property 'ApprovalCheck'
Generate property 'ApprovalCheck'
Generate field 'ApprovalCheck'
Generate read-only field 'ApprovalCheck'
Generate local 'ApprovalCheck'
Generate parameter 'ApprovalCheck'
Solution Ex...
Search Solution
Solution 'Worksh
Assembly-CSI
Git Cha...
Solutio...
Properties






```
1  using System;
2
3  [Serializable]
   0 references
4  public class UserData
5  {
6      public string userName;
7  }
```

```
1   using System;
2   using System.Collections;
3   using System.Collections.Generic;
4   using Unity.Netcode;
5   using UnityEngine;
6
        1 reference
7   public class NetworkServer
8   {
9       private NetworkManager networkManager;
10
            0 references
11      public NetworkServer(NetworkManager networkManager)
12      {
13          this.networkManager = networkManager;
14
15          networkManager.ConnectionApprovalCallback += ApprovalCheck;
16      }
17
            1 reference
18      private void ApprovalCheck(
19          NetworkManager.ConnectionApprovalRequest request,
20          NetworkManager.ConnectionApprovalResponse response)
21      {
22          string payload = System.Text.Encoding.UTF8.GetString(request.Payload);
23          UserData userData = JsonUtility.FromJson<UserData>(payload);
24
25          Debug.Log(userData.userName);
26
27          response.Approved = true;
28      }
29  }
```

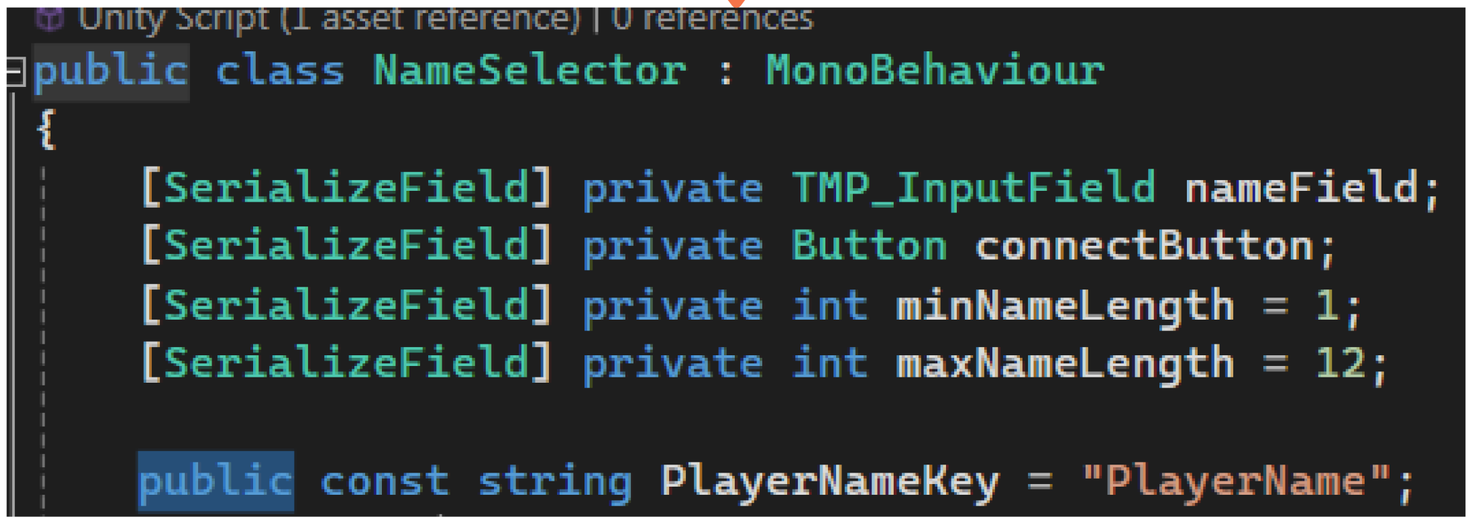
HostGameManager

```
16  public class HostGameManager
17  {
18      private Allocation allocation;
19      private string joinCode;
20      private string lobbyId;
21
22      private NetworkServer networkServer;
23
24      private const int MaxConnections = 20;
25      private const string GameSceneName = "Game";
        1 reference
```

```
79          networkServer = new NetworkServer(NetworkManager.Singleton);
80
81          NetworkManager.Singleton.StartHost();
82
83          NetworkManager.Singleton.SceneManager.LoadScene(GameSceneName, LoadSceneMode.Single);
84      }
85
```

NameSelector

```
    private const string PlayerNameKey = "PlayerName";
```



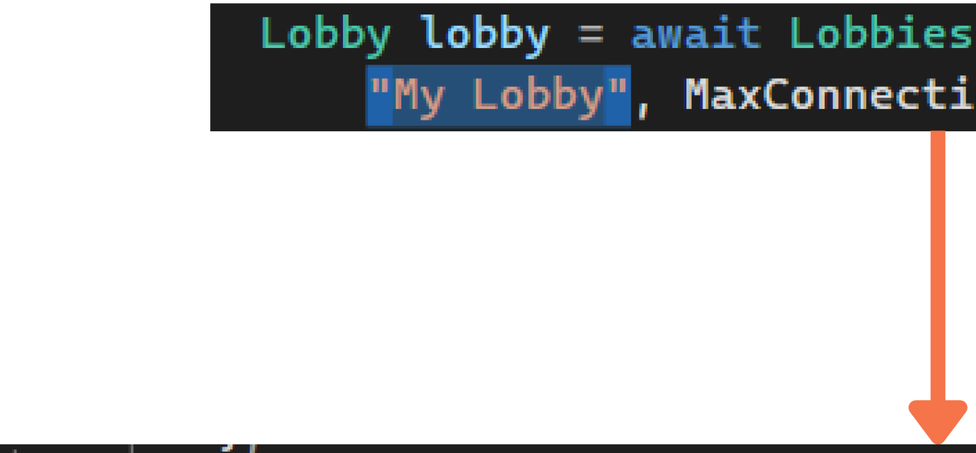
```
Unity Script (1 asset reference) | 0 references
public class NameSelector : MonoBehaviour
{
    [SerializeField] private TMP_InputField nameField;
    [SerializeField] private Button connectButton;
    [SerializeField] private int minNameLength = 1;
    [SerializeField] private int maxNameLength = 12;

    public const string PlayerNameKey = "PlayerName";
```

ClientGameManager

```
35
    2 references
36  public async Task StartClientAsync(string joinCode)
37  {
38      try
39      {
40          allocation = await Relay.Instance.JoinAllocationAsync(joinCode);
41      }
42      catch (Exception e)
43      {
44          Debug.Log(e);
45          return;
46      }
47
48      UnityTransport transport = NetworkManager.Singleton.GetComponent<UnityTransport>();
49
50      RelayServerData relayServerData = new RelayServerData(allocation, "dtls");
51      transport.SetRelayServerData(relayServerData);
52
53      UserData userData = new UserData
54      {
55          userName = PlayerPrefs.GetString(NameSelector.PlayerNameKey, "Missing Name")
56      };
57      string payload = JsonUtility.ToJson(userData);
58      byte[] payloadBytes = Encoding.UTF8.GetBytes(payload);
59
60      NetworkManager.Singleton.NetworkConfig.ConnectionData = payloadBytes;
61
62      NetworkManager.Singleton.StartClient();
63  }
64 }
```

HostGameManager

```
Lobby lobby = await Lobbies.Instance.CreateLobbyAsync
    "My Lobby", MaxConnections, lobbyOptions);
```



```
67      string playerName = PlayerPrefs.GetString(NameSelector.PlayerNameKey, "Unknown");
68      Lobby lobby = await Lobbies.Instance.CreateLobbyAsync(
69          $"{playerName}'s Lobby", MaxConnections, lobbyOptions);
```

ClientGameManager

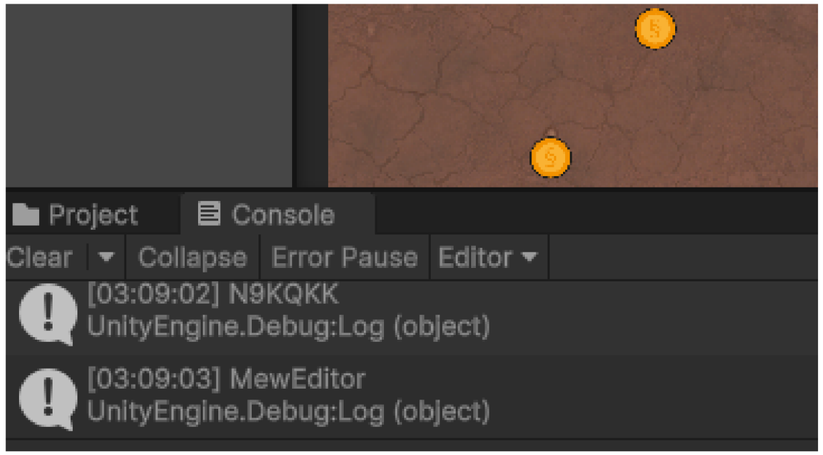
```
tworkServer.cs | ClientGameManager.cs | ClientSingleton.cs | ApplicationController.cs | HostGameManager.cs | NameSelector.
ClientGameManager | StartClientAsync(string joinCode)

        UserData userData = new UserData
        {
            userName = PlayerPrefs.GetString(NameSelector.PlayerNameKey, "Missing Name")
        };
        string payload = JsonUtility.ToJson(userData);
        byte[] payloadBytes = Encoding.UTF8.GetBytes(payload);

        NetworkManager.Singleton.NetworkConfig.ConnectionData = payloadBytes;

        NetworkManager.Singleton.StartClient();
    }
}
```

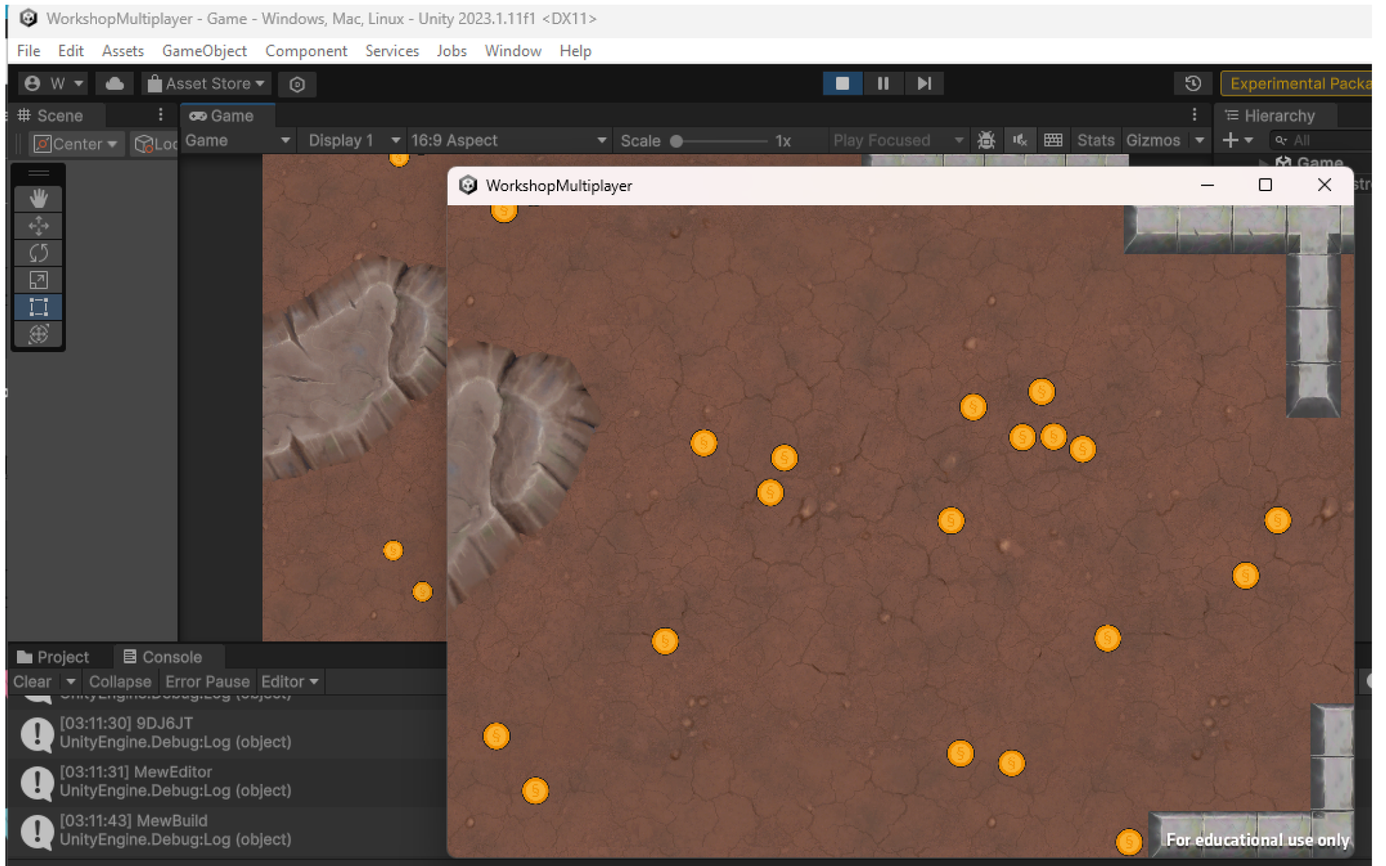
HostGameManager

```
rkServer.cs | HostGameManager.cs* | ClientGameManager.cs | ClientSingleton.cs | ApplicationController.cs | NameSelector.cs
HostGameManager | StartHostAsync()

            return;
        }

        networkServer = new NetworkServer(NetworkManager.Singleton);

        UserData userData = new UserData
        {
            userName = PlayerPrefs.GetString(NameSelector.PlayerNameKey, "Missing Name")
        };
        string payload = JsonUtility.ToJson(userData);
        byte[] payloadBytes = Encoding.UTF8.GetBytes(payload);

        NetworkManager.Singleton.NetworkConfig.ConnectionData = payloadBytes;

        NetworkManager.Singleton.StartHost();

        NetworkManager.Singleton.SceneManager.LoadScene(GameSceneName, LoadSceneMode.Single);
    }
```


MewEditor
Connect


MewEditor's Lobby
1/20
Join
Refresh


Project
Console
Clear
Collapse
Error Pause
Editor
[03:09:02] N9KQKK
UnityEngine.Debug:Log (object)
[03:09:03] MewEditor
UnityEngine.Debug:Log (object)


WorkshopMultiplayer - Game - Windows, Mac, Linux - Unity 2023.1.11f1 <DX11>
File
Edit
Assets
GameObject
Component
Services
Jobs
Window
Help
Asset Store
Scene
Game
Hierarchy
Display 1
16:9 Aspect
Scale
1x
Play Focused
Stats
Gizmos
WorkshopMultiplayer
Project
Console
Clear
Collapse
Error Pause
Editor
[03:11:30] 9DJ6JT
UnityEngine.Debug:Log (object)
[03:11:31] MewEditor
UnityEngine.Debug:Log (object)
[03:11:43] MewBuild
UnityEngine.Debug:Log (object)
For educational use only

NetworkServer.cs

```
1 reference
private void ApprovalCheck(
    NetworkManager.ConnectionApprovalRequest request,
    NetworkManager.ConnectionApprovalResponse response)
{
    string payload = System.Text.Encoding.UTF8.GetString(request.Payload);
    UserData userData = JsonUtility.FromJson<UserData>(payload);

    Debug.Log(userData.userName);

    response.Approved = true;
    response.CreatePlayerObject = true;
}
```


WorkshopMultiplayer
Project
Console
Clear
Collapse
Error Pause
Editor
[03:19:54] LM7W9G
UnityEngine.Debug:Log (object)
[03:19:55] MewEditor
UnityEngine.Debug:Log (object)
[03:20:16] MewBuild
UnityEngine.Debug:Log (object)
For educational use only

Handling Connections

เนื้อหาในส่วนนี้เราจะทำการจัดการ Connection
หนึ่งในนั้นก็คือการเก็บค่า ClientID ของ Player ที่เข้ามาไว้

และเราจะจัดการทั้งฝั่ง Connections และ Disconnections
เพื่อที่ทำหน้าที่เก็บข้อมูล และ ละทิ้งข้อมูลออกไปได้

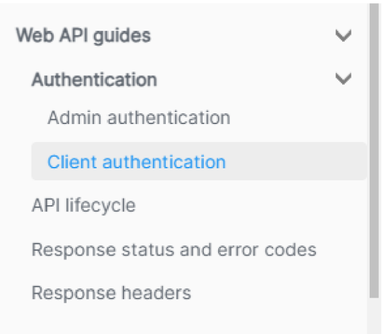
```cs
using System;

[Serializable]
6 references
public class UserData
{
    public string userName;
    public string userAuthId;
}
```

เมื่อมีการเพิ่ม Data ใหม่เข้ามา ลำดับต่อไป

จำเป็นต้อง Setup ค่าเข้าไปใน UserData ที่ถูก new ด้วย

“ ClientGameManager.cs & HostGameManager.cs ”

```cs
using Unity.Services.Authentication;
UserData userData = new UserData
{
    userName = PlayerPrefs.GetString(NameSelector.PlayerNameKey, "Missing Name"),
    userAuthId = AuthenticationService.Instance.PlayerId
};
```



Set up the Unity Authentication service

When a new or returning user logs into your application, the Unity Authentication service uses the following to capture the user's information:

- `PlayerId`: the main identifier for new and returning Unity Authentication end users.
- `sessionToken`: a randomly-generated string that re-authenticates the user after the session expires.
- `idToken`: a JSON Web Token (JWT) that contains a set of claims, including the `PlayerId`. The `idToken` provides authorization when passed into other Unity Services APIs.

Client authentication | Unity Services Web API docs

Client APIs are endpoints that allow you to access Unity Services as a Unity project user. These endpoints are called inside the project after

unity.com

```
1 reference
private void ApprovalCheck(
    NetworkManager.ConnectionApprovalRequest request,
    NetworkManager.ConnectionApprovalResponse response)
{
    string payload = System.Text.Encoding.UTF8.GetString(request.Payload);
    UserData userData = JsonUtility.FromJson<UserData>(payload);

    request.ClientNetworkId
    Debug.Log(userData.userName);

    response.Approved = true;
    response.CreatePlayerObject = true;
}
```

Storing Player Data

- Create 2 dictionaries, **<ulong, string>** and **<string, UserData>**
- Use **request.ClientNetworkId** as the key to set the **userData.userAuthId** value
- Use the **userData.userAuthId** as the key to set the **userData** value

NetworkServer.cs

```cs
using System;
using System.Collections;
using System.Collections.Generic;
using Unity.Netcode;
using UnityEngine;

3 references
public class NetworkServer
{
    private NetworkManager networkManager;

    private Dictionary<ulong,string> clientIdToAuth = new Dictionary<ulong,string>();
    private Dictionary<string,UserData> authIdToUserData = new Dictionary<string,UserData>();
```

```cs
1 reference
private void ApprovalCheck(
    NetworkManager.ConnectionApprovalRequest request,
    NetworkManager.ConnectionApprovalResponse response)
{
    string payload = System.Text.Encoding.UTF8.GetString(request.Payload);
    UserData userData = JsonUtility.FromJson<UserData>(payload);

    clientIdToAuth[request.ClientNetworkId] = userData.userAuthId;
    authIdToUserData[userData.userAuthId] = userData;
    //Debug.Log(userData.userName);

    response.Approved = true;
    response.CreatePlayerObject = true;
}
```

การตั้งค่ารูปแบบนี้เพื่อที่จะเก็บข้อมูลของผู้เล่นไว้ใน Dictionary โดย authIdToUserData จะเก็บ userData ผูกไว้กับ userAuthId วิธีการในรูปแบบนี้เพื่อความสะดวกในการเรียกข้อมูลของ userData มาใช้ในอนาคตได้

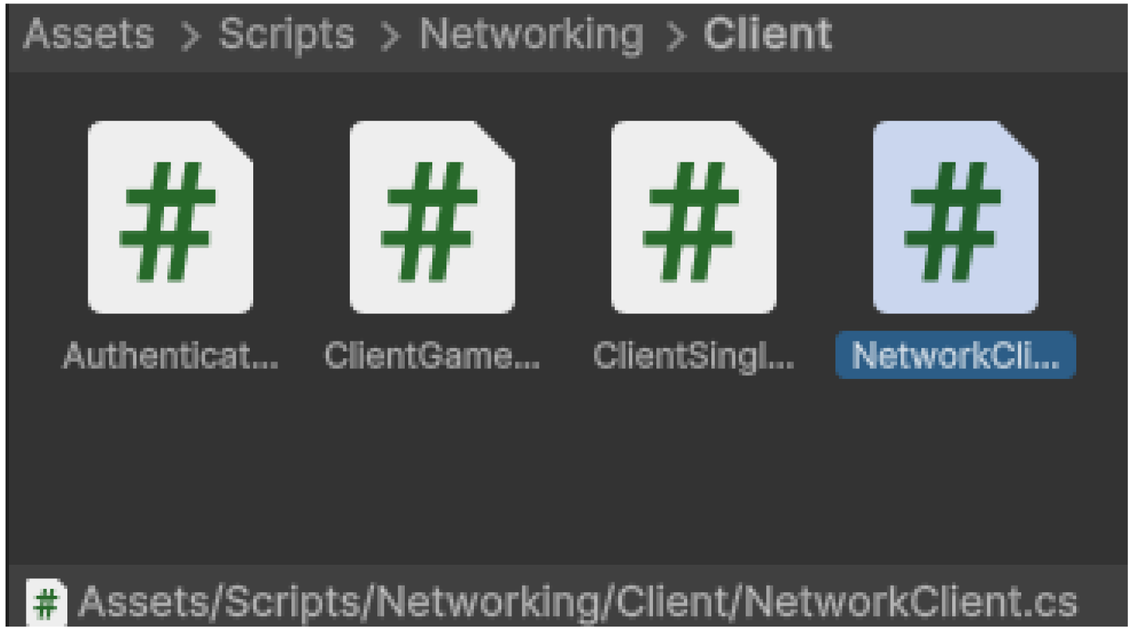
เช่น การเลือกเปลี่ยนสีของตัวละครก่อนเข้าเกม เป็นต้น

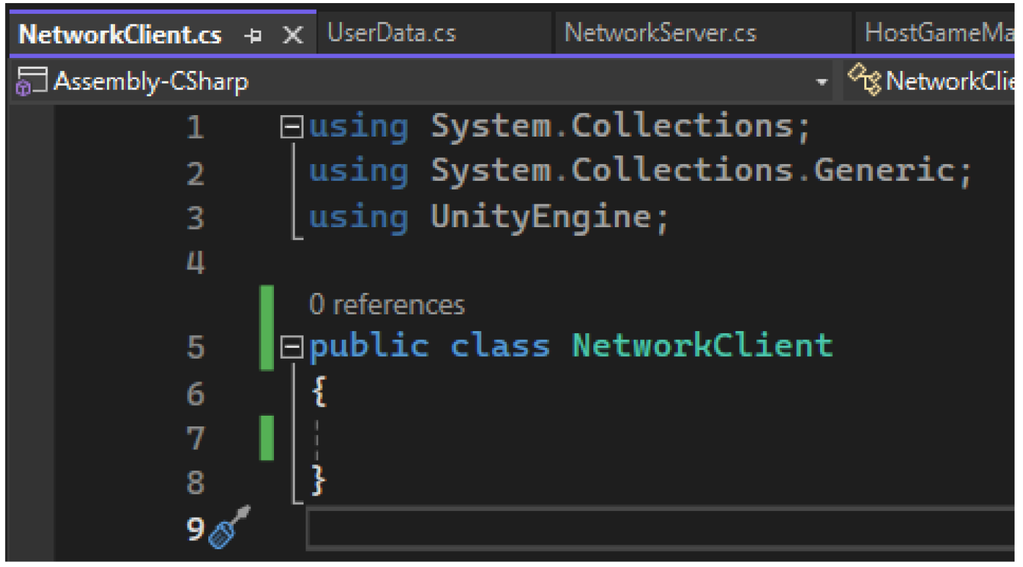
NetworkServer.cs

```
public NetworkServer(NetworkManager networkManager)
{
    this.networkManager = networkManager;

    networkManager.ConnectionApprovalCallback += ApprovalCheck;
    networkManager.OnServerStarted += OnNetworkReady;
}
```

```
private void OnNetworkReady()
{
    networkManager.OnClientDisconnectCallback += OnClientDisconnect;
}
```

```
private void OnClientDisconnect(ulong clientId)
{
    if(clientIdToAuth.TryGetValue(clientId,out string authId))
    {
        clientIdToAuth.Remove(clientId);
        authIdToUserData.Remove(authId);
    }
}
```

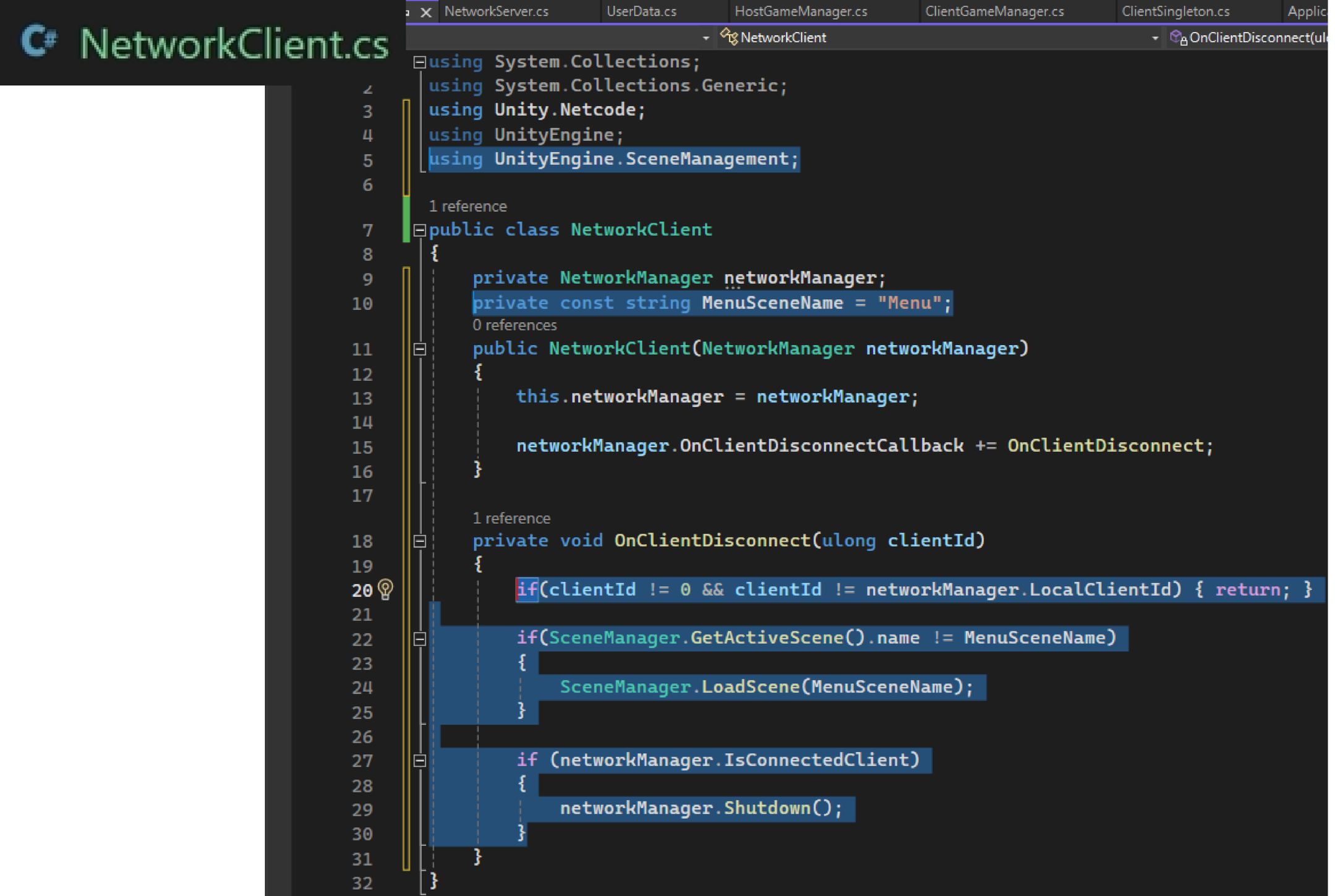

Assets > Scripts > Networking > Client
Authenticat...
ClientGame...
ClientSingl...
NetworkCli...
Assets/Scripts/Networking/Client/NetworkClient.cs


NetworkClient.cs
UserData.cs
NetworkServer.cs
HostGameMa
Assembly-CSharp
NetworkClie
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
0 references
public class NetworkClient
{
}

NetworkClient.cs

```cs
using System.Collections;
using System.Collections.Generic;
using Unity.Netcode;
using UnityEngine;

1 reference
public class NetworkClient
{
    private NetworkManager networkManager;

    0 references
    public NetworkClient(NetworkManager networkManager)
    {
        this.networkManager = networkManager;

        networkManager.OnClientDisconnectCallback += OnClientDisconnect;
    }

    1 reference
    private void OnClientDisconnect(ulong clientId)
    {

    }
}

```

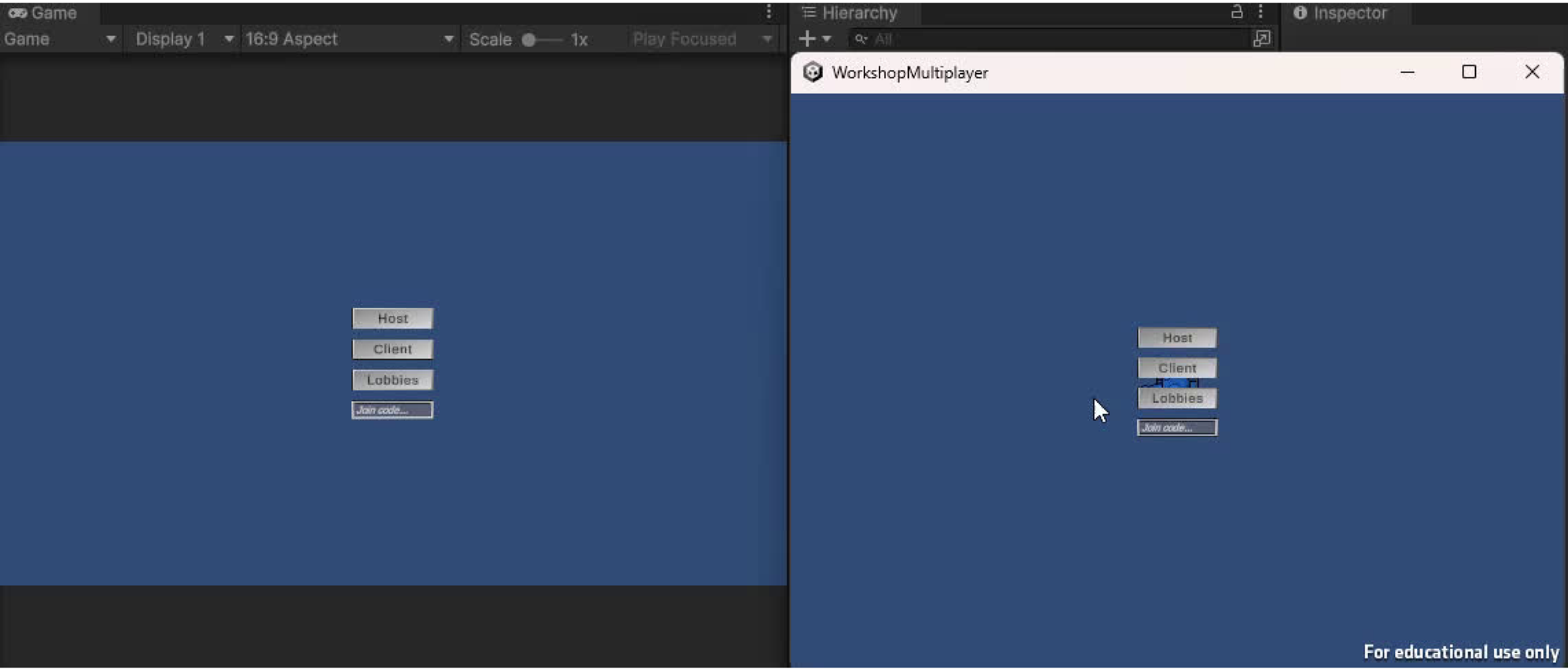
NetworkClient.cs




```
using System.Collections;
using System.Collections.Generic;
using Unity.Netcode;
using UnityEngine;
using UnityEngine.SceneManagement;

1 reference
public class NetworkClient
{
    private NetworkManager networkManager;
    private const string MenuSceneName = "Menu";
    0 references
    public NetworkClient(NetworkManager networkManager)
    {
        this.networkManager = networkManager;

        networkManager.OnClientDisconnectCallback += OnClientDisconnect;
    }

    1 reference
    private void OnClientDisconnect(ulong clientId)
    {
        if(clientId != 0 && clientId != networkManager.LocalClientId) { return; }

        if(SceneManager.GetActiveScene().name != MenuSceneName)
        {
            SceneManager.LoadScene(MenuSceneName);
        }

        if (networkManager.IsConnectedClient)
        {
            networkManager.Shutdown();
        }
    }
}
```

ClientGameManager.cs

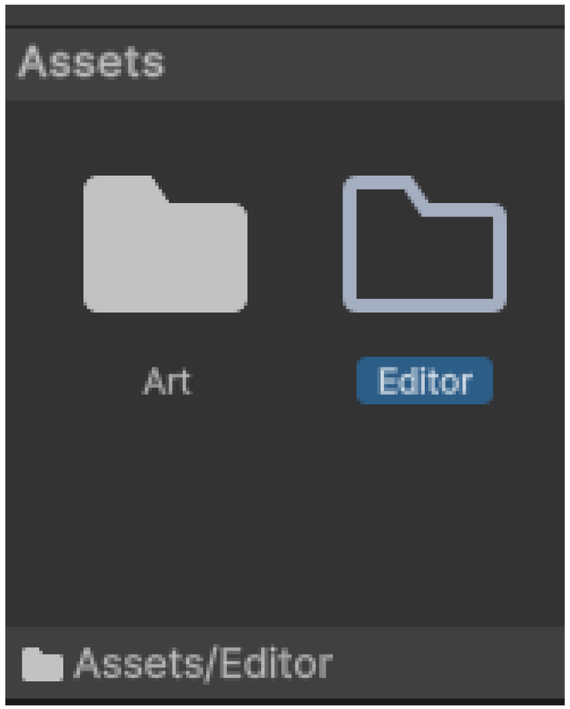
```
public class ClientGameManager
{
    private JoinAllocation allocation;
    private NetworkClient networkClient;

    private const string MenuSceneName = "Menu";
    1 reference
    public async Task<bool> InitAsync()
    {
        await UnityServices.InitializeAsync();

        networkClient = new NetworkClient(NetworkManager.Singleton);

        AuthState authState = await AuthenticationWrapper.DoAuth();
        if(authState == AuthState.Authenticated)
        {
            return true;
        }
        return false;
    }
```

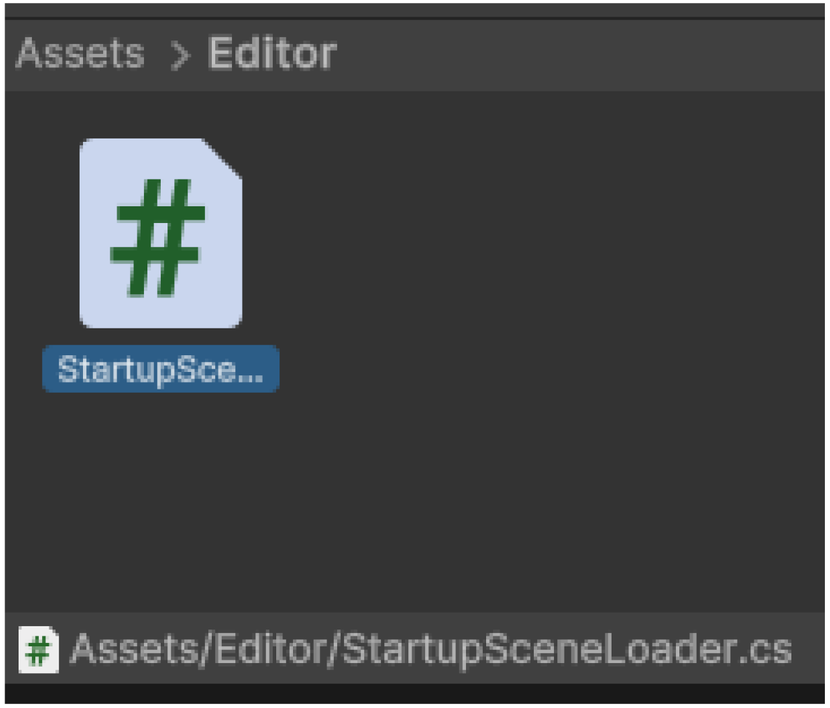


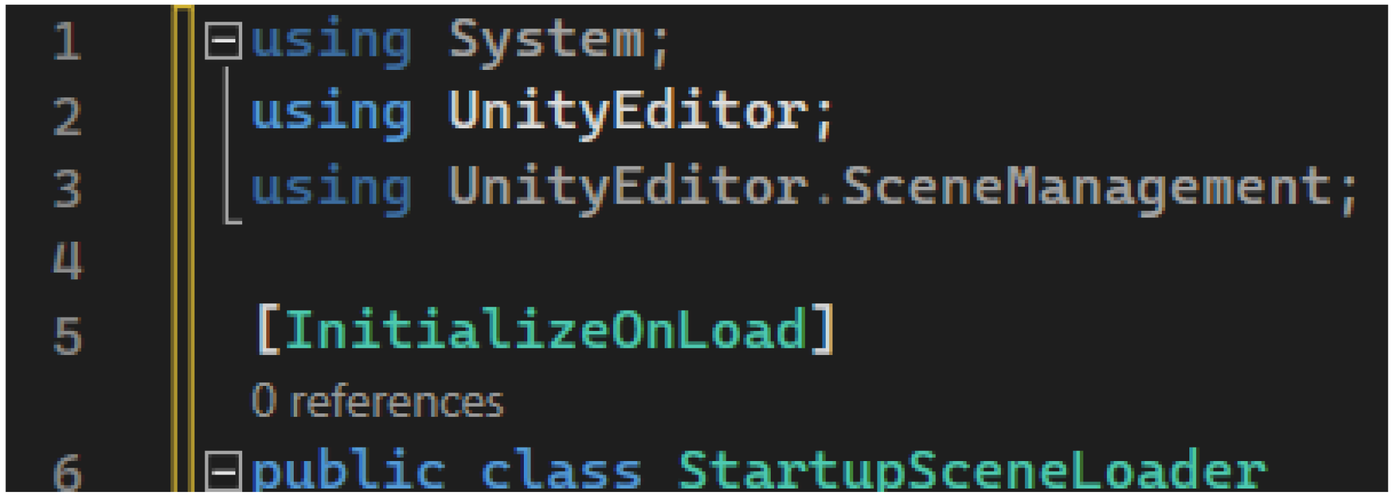
ในตอนนี้ฝั่ง Client ก็จะเด้งออกมาหน้า Menu โดยอัตโนมัติเมื่อ Server ที่เข้าอยู่ปิดตัวหรือหลุดได้
ส่วน userData ต่างๆ ที่เก็บไว้จะเอามาใช้ต่อในบทต่อๆ ไป

Networking Improvements

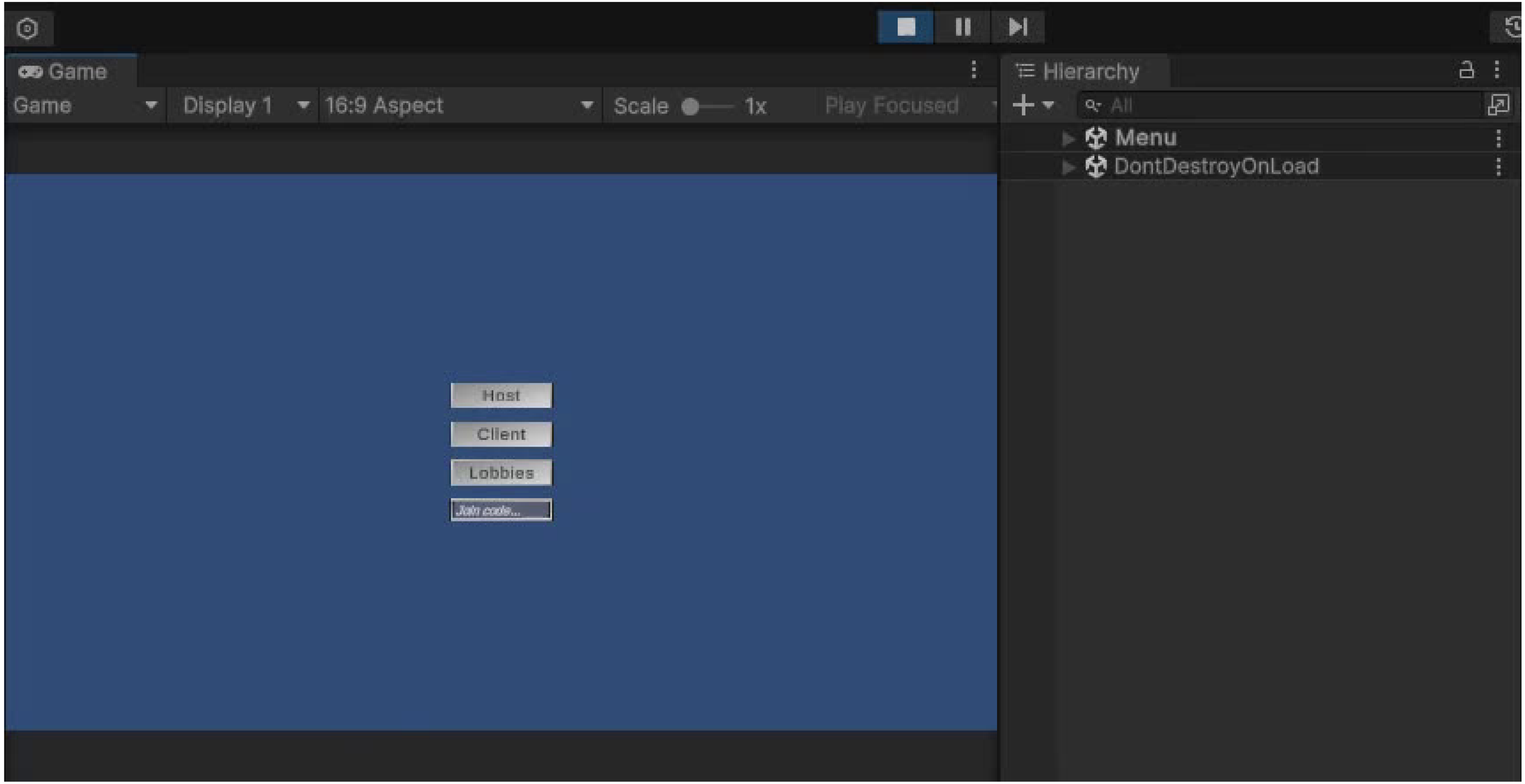
ในบทนี้เราจะใช้โค้ด Editor จัดการ Startup Scene Loader เพื่อให้ง่ายต่อการทดสอบ โดยไม่ต้องเสียเวลากดเลือก Scene จากใน Project


Assets
Art
Editor
Assets/Editor


Assets > Editor
StartupSce...
Assets/Editor/StartupSceneLoader.cs


1 using System;
2 using UnityEditor;
3 using UnityEditor.SceneManagement;
4
5 [InitializeOnLoad]
0 references
6 public class StartupSceneLoader

```
using System;
using UnityEditor;
using UnityEditor.SceneManagement;

[InitializeOnLoad]
1 reference
public class StartupSceneLoader
{
    0 references
    static StartupSceneLoader()
    {
        EditorApplication.playModeStateChanged += LoadStartupScene;
    }

    1 reference
    private static void LoadStartupScene(PlayModeStateChange state)
    {
        if(state == PlayModeStateChange.ExitingEditMode)
        {
            EditorSceneManager.SaveCurrentModifiedScenesIfUserWantsTo();
        }

        if(state == PlayModeStateChange.EnteredPlayMode)
        {
            if(EditorSceneManager.GetActiveScene().buildIndex != 0)
            {
                EditorSceneManager.LoadScene(0);
            }
        }
    }
}
```



เราจะเห็นว่าไม่ว่าจะเปิดจากฉากไหน Unity Editor จะบังคับ
ให้มาเปิดตั้งแต่ฉากแรกเองโดยอัตโนมัติ

Shutting Down Cleanly

ในบทนี้เราจะมา Clean กันครับ จะมาจัดการพวกที่เรา Subscibed และ ที่ใช้ Service ต่างๆ ไว้ ในกรณีที่ Shutdown server ทั้งจากการออกแบบปกติ และไม่ปกติ เพื่อแก้ประเด็น Unexpected Errors ที่จะเกิดขึ้นได้

IDisposable Interface

Reference

Feedback

Definition

Namespace: System

Assembly: System.Runtime.dll

Provides a mechanism for releasing unmanaged resources.

C# Copy

```
public interface IDisposable
```

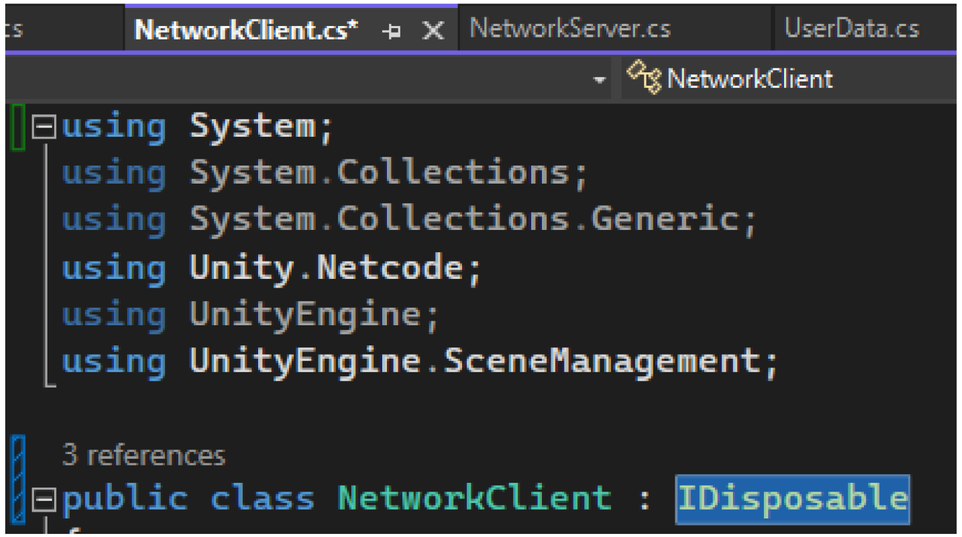
Derived Microsoft.Extensions.Caching.Memory.ICacheEntry
Microsoft.Extensions.Caching.Memory.IMemoryCache
Microsoft.Extensions.Caching.Memory.MemoryCache
Microsoft.Extensions.Caching.Redis.RedisCache
Microsoft.Extensions.Caching.StackExchangeRedis.RedisCache
More...

IDisposable Interface (System)

Provides a mechanism for releasing unmanaged resources.

MicrosoftLearn /

NetworkClient.cs




```
public void Dispose()
{
    if(networkManager != null)
    {
        networkManager.OnClientDisconnectCallback -= OnClientDisconnect;
    }
}
```

ClientGameManager.cs

```
16  public class ClientGameManager : IDisposable
```

```
public void Dispose()
{
    networkClient?.Dispose();
}
```

ClientSingleton.cs

```
private void OnDestroy()
{
    GameManager?.Dispose();
}
```

NetworkServer.cs

```
1   using System;
2   using System.Collections;
3   using System.Collections.Generic;
4   using Unity.Netcode;
5   using UnityEngine;
6
        3 references
7   public class NetworkServer : IDisposable
```

```
public void Dispose()
{
    if(networkManager == null) { return; }

    networkManager.ConnectionApprovalCallback -= ApprovalCheck;
    networkManager.OnClientDisconnectCallback -= OnClientDisconnect;
    networkManager.OnServerStarted -= OnNetworkReady;

    if (networkManager.IsListening)
    {
        networkManager.Shutdown();
    }
}
```

HostGameManager.cs

```
18  public class HostGameManager : IDisposable
```

```
public async void Dispose()
{
    HostSingleton.Instance.StopCoroutine(nameof(HeartbeatLobby));

    if(!string.IsNullOrEmpty(lobbyId) )
    {
        try
        {
            await Lobbies.Instance.DeleteLobbyAsync(lobbyId);
        }
        catch(LobbyServiceException e)
        {
            Debug.Log(e);
        }

        lobbyId = string.Empty;
    }

    networkServer?.Dispose();
}
```

HostSingleton.cs

```
private void OnDestroy()
{
    GameManager?.Dispose();
}
```

ในตอนนี้ตอนที่เรากดปิด หรือหยุด server กะทันหันจะไม่เห็น
มี Warning หรือ Error อะไรมาขึ้นเตือนอีกต่อไป เพราะเรา
ได้ทำการ Clean เป็นที่เรียบร้อยแล้วนั่นเอง

WorkshopMultiplayer\Assets\Scripts\Networking\Client**ClientGameManager.cs**

```
@@ -13,7 +13,7 @@
13 13   using System.Text;
14 14   using Unity.Services.Authentication;
15 15
16    - public class ClientGameManager
   16 + public class ClientGameManager : IDisposable
17 17   {
18 18       private JoinAllocation allocation;
19 19       private NetworkClient networkClient;
@@ -67,4 +67,9 @@ public async Task StartClientAsync(string joinCode)
67 67
68 68           NetworkManager.Singleton.StartClient();
69 69       }
   70 +
   71 +     public void Dispose()
   72 +     {
   73 +         networkClient?.Dispose();
   74 +     }
70 75   }
```

WorkshopMultiplayer\Assets\Scripts\Networking\Client**ClientSingleton.cs**

```
@@ -36,4 +36,9 @@ public async Task<bool> CreateClient()
36 36
37 37           return await GameManager.InitAsync();
38 38       }
   39 +
   40 +     private void OnDestroy()
   41 +     {
   42 +         GameManager?.Dispose();
   43 +     }
39 44   }
```

WorkshopMultiplayer\Assets\Scripts\Networking\Client**NetworkClient.cs**

```
@@ -1,10 +1,11 @@
   1 + using System;
1  2   using System.Collections;
2  3   using System.Collections.Generic;
3  4   using Unity.Netcode;
4  5   using UnityEngine;
5  6   using UnityEngine.SceneManagement;
6  7
7     - public class NetworkClient
   8 + public class NetworkClient : IDisposable
8  9   {
9  10      private NetworkManager networkManager;
10 11      private const string MenuSceneName = "Menu";
@@ -29,4 +30,12 @@ private void OnClientDisconnect(ulong clientId)
29 30              networkManager.Shutdown();
30 31          }
31 32      }
   33 +
   34 +     public void Dispose()
   35 +     {
   36 +         if(networkManager != null)
   37 +         {
   38 +             networkManager.OnClientDisconnectCallback -= OnClientDisconnect;
   39 +         }
   40 +     }
32 41  }
```

WorkshopMultiplayer\Assets\Scripts\Networking\Host**HostGameManager.cs**

```
+  public class HostGameManager : IDisposable
   {
      private Allocation allocation;
      private string joinCode;

@@ -105,4 +105,25 @@ private IEnumerator HeartbeatLobby(float waitTimeSeconds)
                yield return delay;
            }
        }
+
+     public async void Dispose()
+     {
+         HostSingleton.Instance.StopCoroutine(nameof(HeartbeatLobby));
+
+         if(!string.IsNullOrEmpty(lobbyId) )
+         {
+             try
+             {
+                 await Lobbies.Instance.DeleteLobbyAsync(lobbyId);
+             }
+             catch(LobbyServiceException e)
+             {
+                 Debug.Log(e);
+             }
+
+             lobbyId = string.Empty;
+         }
+
+         networkServer?.Dispose();
+     }
   }
```

WorkshopMultiplayer\Assets\Scripts\Networking\Server**NetworkServer.cs**

```
@@ -4,7 +4,7 @@
   using Unity.Netcode;
   using UnityEngine;

-  public class NetworkServer
+  public class NetworkServer : IDisposable
   {
       private NetworkManager networkManager;

@@ -47,4 +47,18 @@ private void OnClientDisconnect(ulong clientId)
                authIdToUserData.Remove(authId);
            }
        }
+
+     public void Dispose()
+     {
+         if(networkManager == null) { return; }
+
+         networkManager.ConnectionApprovalCallback -= ApprovalCheck;
+         networkManager.OnClientDisconnectCallback -= OnClientDisconnect;
+         networkManager.OnServerStarted -= OnNetworkReady;
+
+         if (networkManager.IsListening)
+         {
+             networkManager.Shutdown();
+         }
+     }
   }
```

Assignment

ให้ทำการถ่ายคลิปผลลัพธ์ของการทำ Workshop ตามเนื้อหา Workshop ในแต่ละหัวข้อนี้พร้อมอธิบายประกอบ

- Player Name Selection
- Connection Approval
- Handling Connections
- Networking Improvements
- Shutting Down Cleanly

https://discord.gg/24xmUFHfzR

WOLVEDEN ACADEMY

THANK YOU

facebook.com/GameDevMew