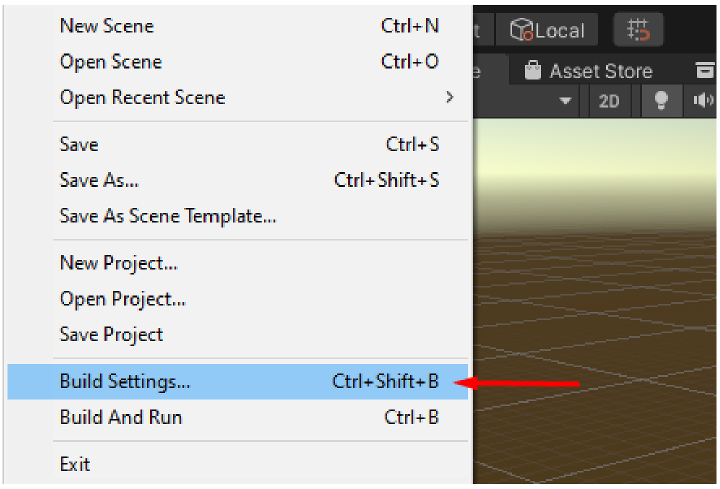
Contents:

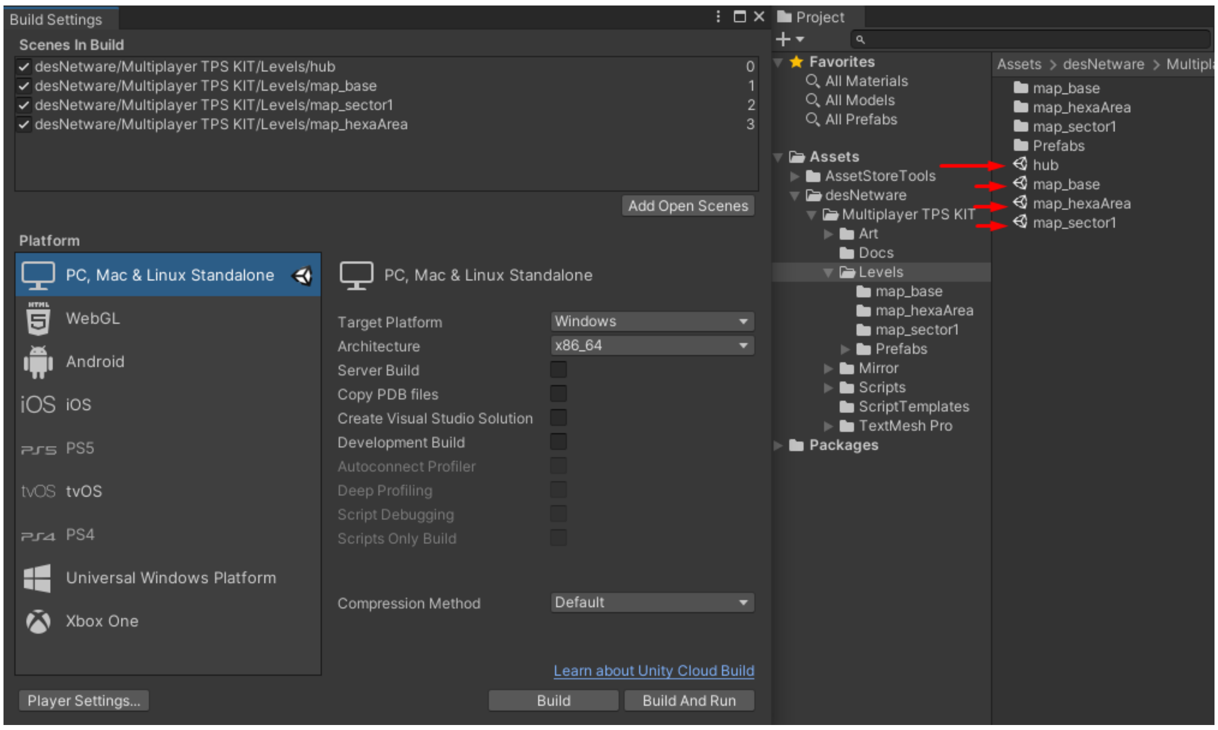
1. Project setup

If You prefer you can also watch Youtube tutorial here

1. After importing MTPSKIT and Mirror, open Build Settings window, go to: File->Build Settings



2. Drag and drop all scenes from Assets\desNetware\Multiplayer TPS KIT. Scene ,,hub" must be the first one, with index 0.



Build Settings

Scenes In Build

✓ desNetware/Multiplayer TPS KIT/Levels/hub	0
✓ desNetware/Multiplayer TPS KIT/Levels/map_base	1
✓ desNetware/Multiplayer TPS KIT/Levels/map_sector1	2
✓ desNetware/Multiplayer TPS KIT/Levels/map_hexaArea	3

Add Open Scenes
Platform
PC, Mac & Linux Standalone
WebGL
Android
iOS
PS5
tvOS
PS4
Universal Windows Platform
Xbox One
PC, Mac & Linux Standalone

Target Platform: Windows
Architecture: x86_64
Server Build
Copy PDB files
Create Visual Studio Solution
Development Build
Autoconnect Profiler
Deep Profiling
Script Debugging
Scripts Only Build
Compression Method: Default

3. Go to: Multiplayer TPS KIT/Setup. Select gameobject ,,GENERATELAYERS" and click ,,Generate Layers and setup script execution order" button in the inspector to generate layers that this kit uses, and set proper execution order for certain scripts. Keep in mind that if You have any layers set in your project, then they will be overwritten.
4. Everything is set, project will automatically load ,,hub'' scene from where you can host your games or connect to them. It makes it also much easier to test your project since you won't have to change

to hub scene every time you made any changes in other scenes. Code responsible for doing that is placed in “Preloader” class, so you can remove/comment it if You wish to, without any harm.

2.Project overview

Project features:

-TPP controller

-Inventory system with ability to pick up and drop items

-3 playable maps

-4 weapons: rifle, pistol, shotgun and sword

-Killfeed with icons for headshots and weapons

-Two gamemodes: Deathmatch and Team Deathmatch

-Chat

-Bots with basic AI that chases nearest enemy and attacks

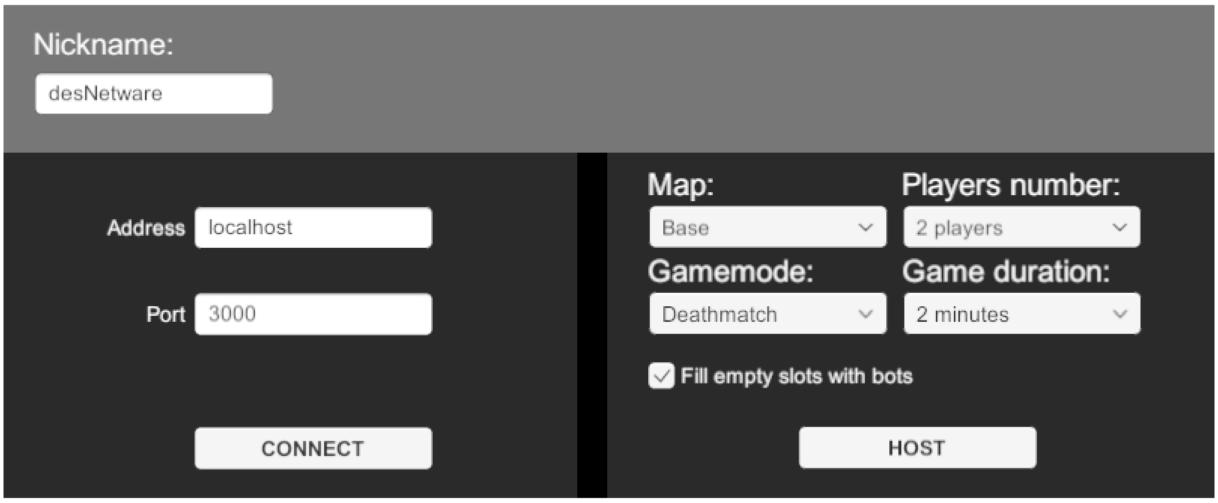
-Simple options menu for changing mouse sensitivity and audio volume

-Object pooler for sparks of bullets hitting objects

3. How MTPS Kit works

(more details can be found in script comments)

1.Game boot


Nickname: desNetware

Address: localhost

Port: 3000

Map: Base

Players number: 2 players

Gamemode: Deathmatch

Game duration: 2 minutes

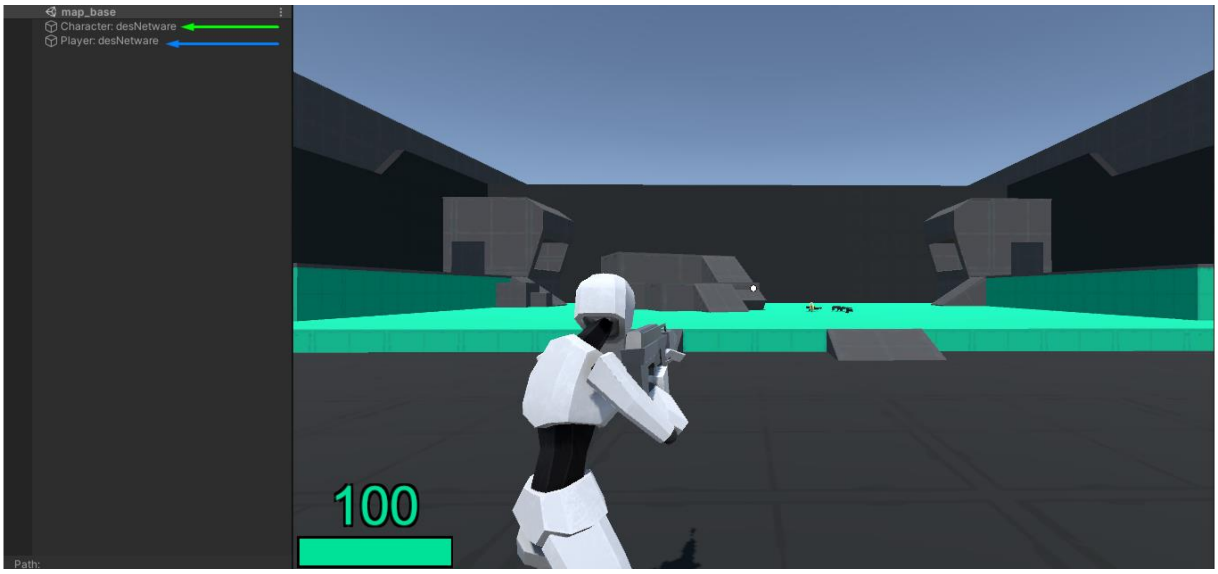
✓ Fill empty slots with bots

When you boot the game You will be given this panel. Here you can host your game or join to existing one. Left part is for connecting and right part is for hosting. Games will be hosted on port typed on the left panel.

You can select map, gamemode (maps can be restricted to provide only certain gamemodes), game duration and max player count.

You can also have bots in your game that will fill empty slots, so server will always be full. If some player will join the game, then one of the bots will “disconnect” to make space for him. When everything is set click “HOST” button to start the game.

2.Players management



Each connected player (and bot) is represented by empty object with component PlayerInstance . This object is named “Player” with player nickname after that. From this objects players can communicate by chat and send spawn requests to the server, which can be accepted or not depending on game state and gamemode. This object exists as long as player is present in the game.

Then, when this object already exists, server will spawn character for given player, if certain conditions for given gamemode and gamestate are met. For example, if we play Team Deathmatch, spawn player after he chooses his team, or respawn player if he died and waited a couple seconds. Or let players spawn only during game, not when game is finished.

3.Character

Every single character You will see in game, You, other player or bot, will use the same character prefab.

Most important character prefab components are:

-CharacterInstance Responsible for main functions of character

-CharacterItemManager Responsible for everything related to the character equipment

4.Item system

Item system consists of two scripts: Item and CharacterItemManager

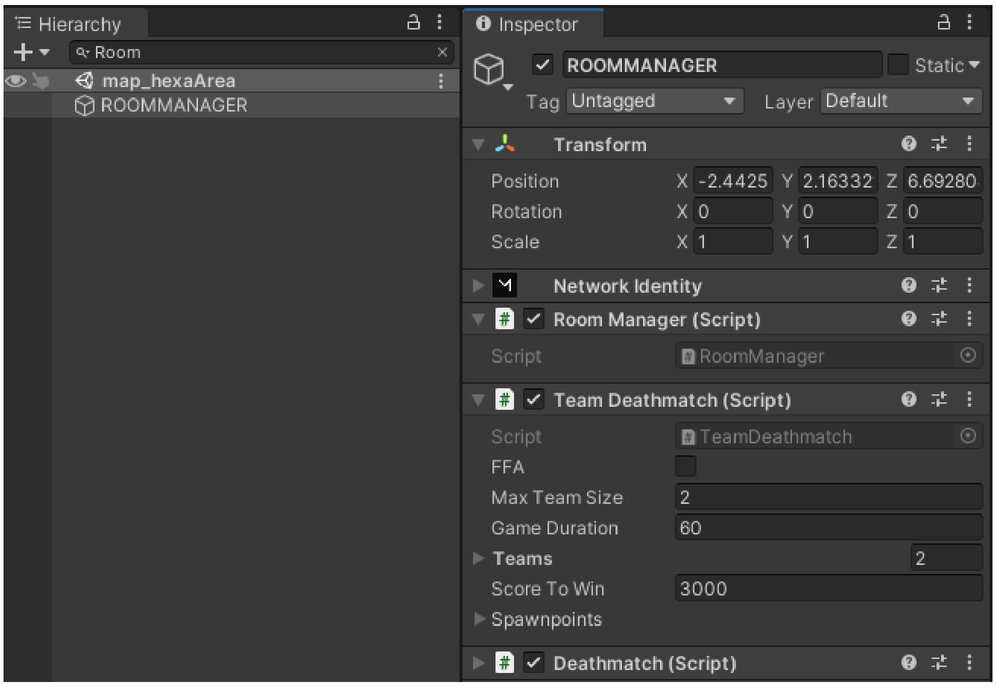
Every single item in the game that is usable for player inherits from Item. This class contains functions for picking, dropping, taking, and putting down item. It also synchronizes this functions for all players connected to the game.

CharacterItemManager manages player equipment. Thanks to this class we can take item, pickup it, drop, drop all of our items on death etc. It is also responsible for synchronization, so if we decide to change weapon, every client will see that.

5.Gamemodes

Gamemode system consists of those scripts: : RoomManager Gamemode and two scripts that inherits from Gamemode: Deathmatch and TeamDeathmatch

All of those scripts are present in map scene, on gameobject “ROOMMANAGER”



Hierarchy
Room
map_hexaArea
ROOMMANAGER

Inspector
ROOMMANAGER Static
Tag Untagged Layer Default

Transform

```
Position   X -2.4425  Y 2.16332  Z 6.69280
Rotation   X 0        Y 0        Z 0
Scale      X 1        Y 1        Z 1
```

Network Identity

Room Manager (Script)
Script RoomManager

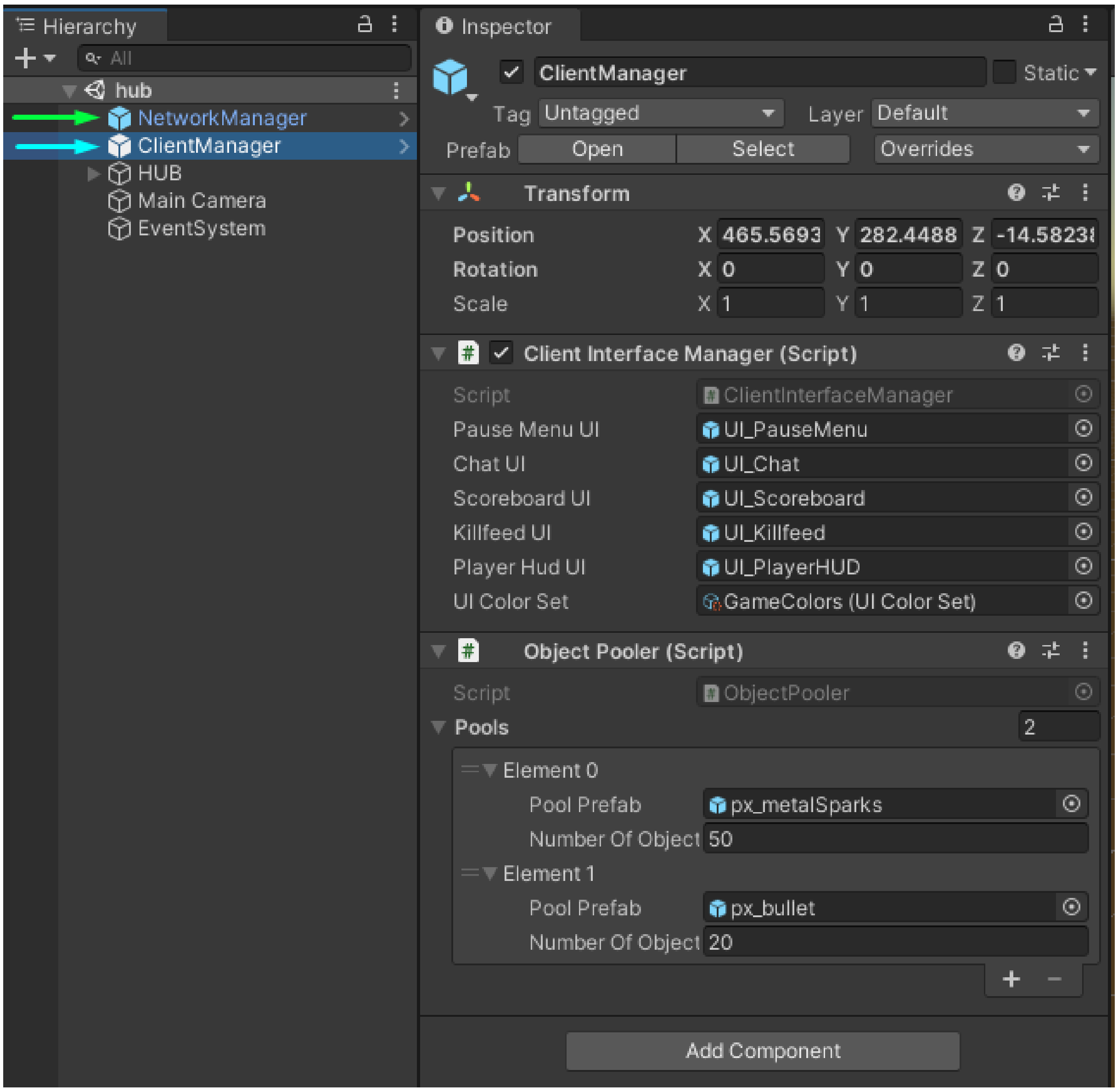
Team Deathmatch (Script)

```
Script          TeamDeathmatch
FFA             ☐
Max Team Size   2
Game Duration   60
Teams           2
Score To Win    3000
Spawnpoints
```

Deathmatch (Script)

Each gamemode is placed in its own component. RoomManager component activates gamemode selected by user. If map supports certain gamemodes, then their components have to be present and set up in this in object “ROOMMANAGER”.

6.Managing prefabs between scenes


Hierarchy
hub
NetworkManager
ClientManager
HUB
Main Camera
EventSystem

Inspector
ClientManager
Static
Tag Untagged
Layer Default
Prefab Open Select Overrides

Transform

Position	X 465.5693	Y 282.4488	Z -14.58238
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Client Interface Manager (Script)

Script	ClientInterfaceManager
Pause Menu UI	UI_PauseMenu
Chat UI	UI_Chat
Scoreboard UI	UI_Scoreboard
Killfeed UI	UI_Killfeed
Player Hud UI	UI_PlayerHUD
UI Color Set	GameColors (UI Color Set)

Object Pooler (Script)

Script	ObjectPooler
Pools	2
Element 0	
Pool Prefab	px_metalSparks
Number Of Object	50
Element 1	
Pool Prefab	px_bullet
Number Of Object	20

Add Component

ClientManager object contains ClientInterfaceManager class which is responsible for Instantiating UI prefabs on gameplay scenes, like scoreboard, player HUD, killfeed and other elements. It is here to avoid placing all of these prefabs on scenes manually. Thanks to that map scenes only need to contain ROOMMANAGER object with gamemodes scripts, which was described in subsection 5, map itself and spawnpoints for players.

NetworkManager object contains CustomNetworkManager class that inherits from NetworkManager class from Mirror networking.

7.Bots

If bot character prefab is spawned server enables on it BaseAI component. It is responsible for bot behaviour. If bot cannot reach enemy or he is too far away, then he will run using navmesh to his target. If target is in shooting range then bot will attack him.

These bots do not use unity component NavMeshAgent. Instead they use exactly the same character controller as players, so they are not stuck to the floor.

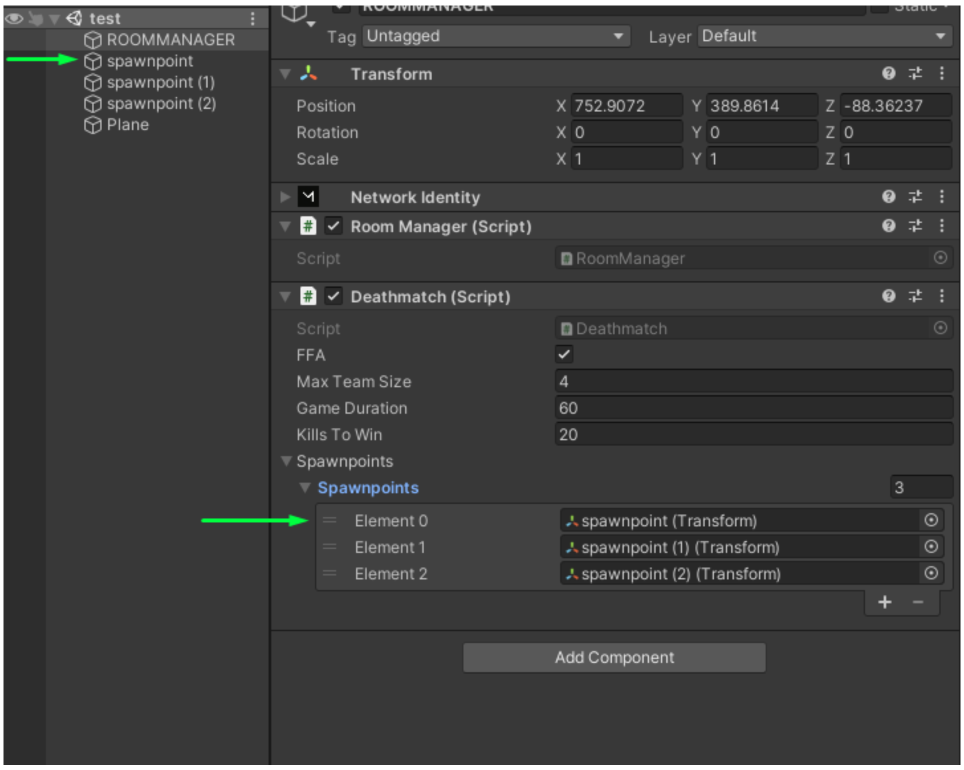
5.How to add another maps

If You prefer you can also watch Youtube tutorial here

1. Create empty scene, save it wherever you want. Create spawnpoints for players as empty gameobjects.

2. Create empty object, add component RoomManager to it, and also gamemode component [Deathmatch TeamDeathmatch] that you wish to have on this map, or both of them.

3. Now we have to setup gamemodes, to do that simply drag and drop previously created spawnpoints objects to your selected gamemode script.


test
ROOMMANAGER
spawnpoint
spawnpoint (1)
spawnpoint (2)
Plane

ROOMMANAGER
Tag Untagged
Layer Default

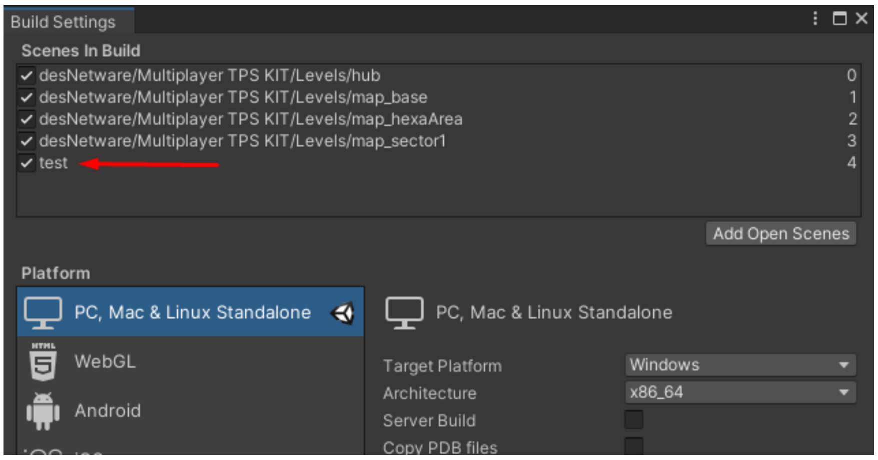
Transform
Position X 752.9072 Y 389.8614 Z -88.36237
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Network Identity

Room Manager (Script)
Script RoomManager

Deathmatch (Script)
Script Deathmatch
FFA ✓
Max Team Size 4
Game Duration 60
Kills To Win 20
Spawnpoints
Spawnpoints 3
Element 0 spawnpoint (Transform)
Element 1 spawnpoint (1) (Transform)
Element 2 spawnpoint (2) (Transform)

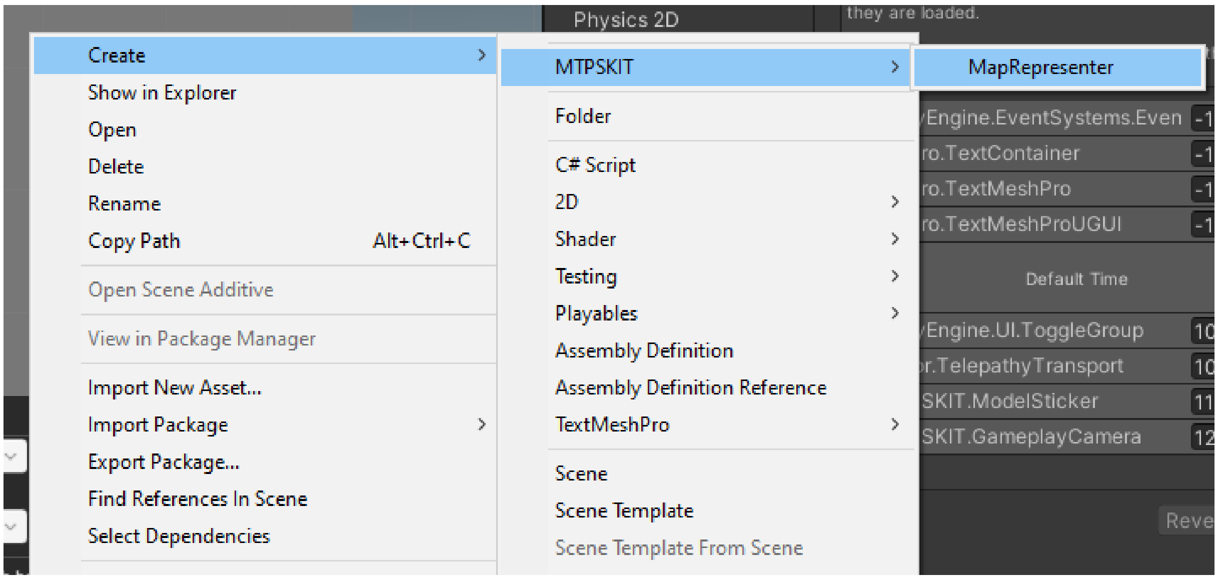
4. Add saved scene to build settings.


Build Settings
Scenes In Build
✓ desNetware/Multiplayer TPS KIT/Levels/hub 0
✓ desNetware/Multiplayer TPS KIT/Levels/map_base 1
✓ desNetware/Multiplayer TPS KIT/Levels/map_hexaArea 2
✓ desNetware/Multiplayer TPS KIT/Levels/map_sector1 3
✓ test 4

PC, Mac & Linux Standalone
Target Platform Windows
Architecture x86_64
Server Build
Copy PDB files

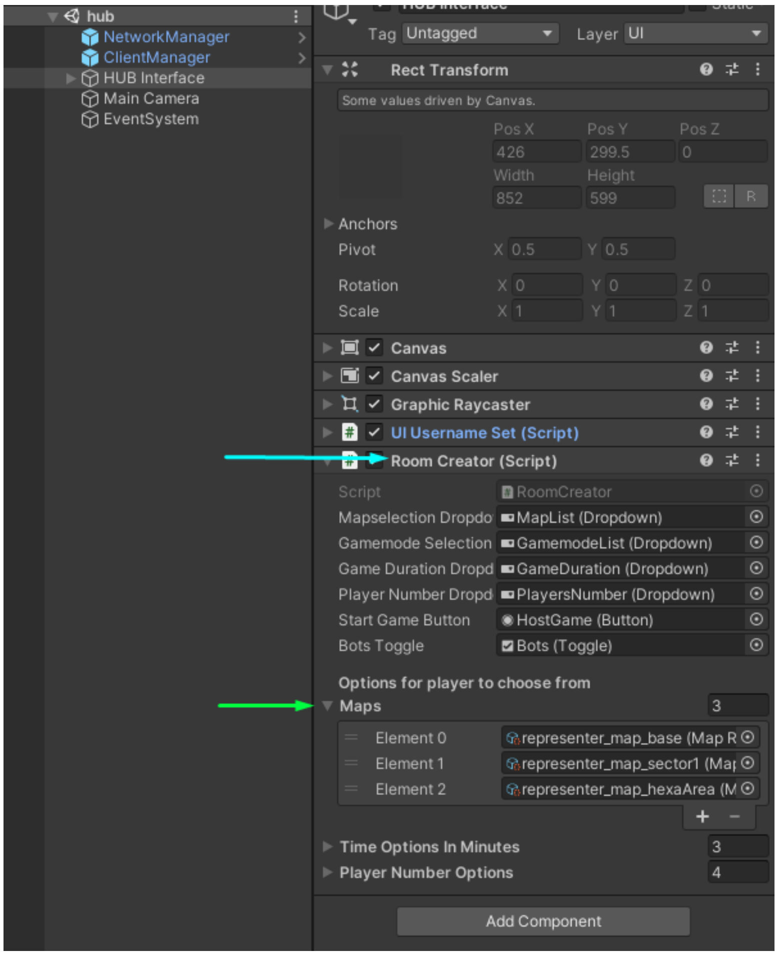
5. Now we have to create ,,map representer" scriptable object, that will tell us in creating room panel on hub scene, how this map is called, and which gamemodes it supports. To do that go to folder you wish to have this map representer object, (representers for our 3 maps are stored in desNetware\Multiplayer TPS KIT\Scripts\MapRepresenters). Right click, point to:

Create->MTPSKIT->MapRepresenter, and name file.



In the inspector of this newly created object drag and drop scene that will be represented by this map representer object, type name of map you wish user creating room will see, and add to AvaibleGamemodes array gamemodes you set up previously making the map.

6.Load ,,hub" scene, select HUB Interface gameobject. In component RoomCreator drag and drop to Maps array your scene representer.



Tag Untagged Layer UI

Pos X	Pos Y	Pos Z
426	299.5	0

Width	Height
852	599

Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Script	RoomCreator
Mapselection Dropdo	MapList (Dropdown)
Gamemode Selection	GamemodeList (Dropdown)
Game Duration Dropd	GameDuration (Dropdown)
Player Number Dropd	PlayersNumber (Dropdown)
Start Game Button	HostGame (Button)
Bots Toggle	Bots (Toggle)
Maps	3
Element 0	representer_map_base (Map R
Element 1	representer_map_sector1 (Ma
Element 2	representer_map_hexaArea (M
Time Options In Minutes	3
Player Number Options	4

7.Done, You can now launch game, select your map, host game and play on it!

Contact:

Email: contact@desnetware.net

Our website

Our Youtube channel

Our AssetStore publisher page

Our Discord server