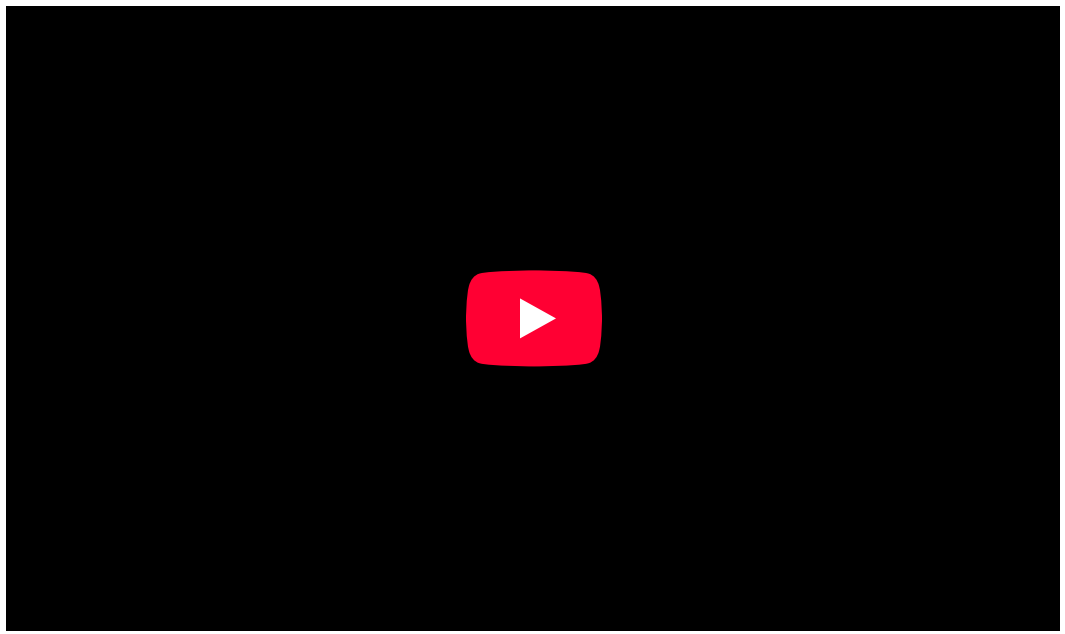
🎓 LESSON 02
Building the Basic Game Field

Start

LESSON GOALS



Video Link

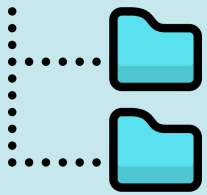
🎯 Goal Summary:

- Build the basic arena layout
- Create scene structure & folders
- Place the stadium model
- Create invisible boundaries
- Add skybox
- Prepare initial GameController script

Next

Learning Objectives + Deliverables

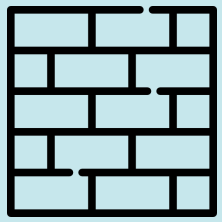
Learning Objectives



Create an organized project/folder structure in Unity

Import and configure a stadium model

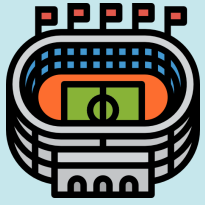
Understand initial spawn positioning



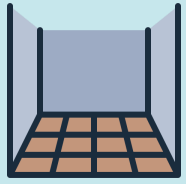
Add invisible collider walls to restrict movement

Implement a basic GameController script that adjusts player movement speed when the player joins the world

Deliverables



✓ A functional arena layout



✓ A transparent, correctly scaled floor

✓ Invisible boundary walls that prevent escape

✓ A functional GameController script modifying player speed

✓ Arena successfully tested in Unity and VRChat

GENERAL DESCRIPTION OF THE LESSON

High-Level Description

🎯 **Goal:** In this exercise we will construct the game field (arena) where the entire VRChat sports game takes place. We will do our very first Udon script **(GameController)**

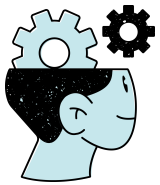
🏐 **LEVEL 1: Advanced Challenge**

In this lesson we will:

- Build the /Game/ folder structure considering that we are going to use design pattern MVC

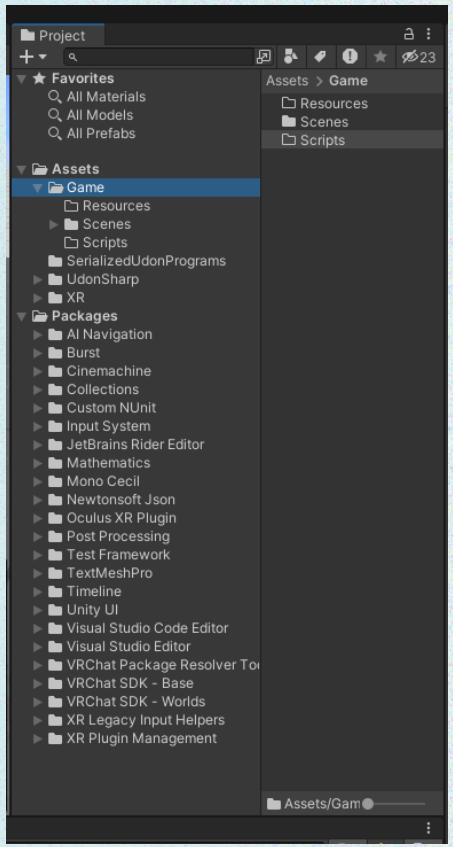
- Place a reference sphere for the center of the game field
- Import and unpack the stadium prefab and convert it into their own
- Add invisible walls to restrict movement
- Add a skybox for atmosphere: Use **(RenderSettings.skybox)**
- Write a small **GameController** script (Udon script) that modifies player's local movement speed

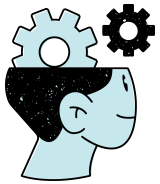
Next



Exercise 1: Create the project folder structure.

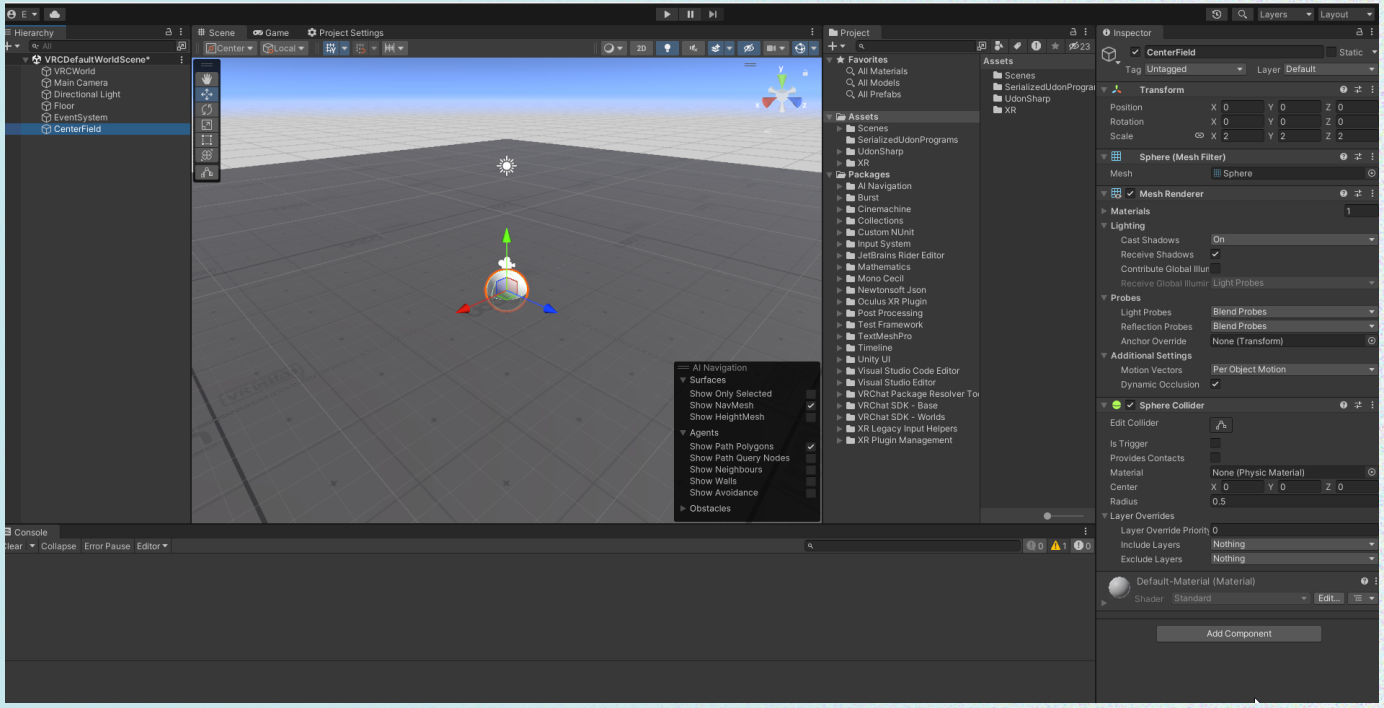
- You should create:
  - /Game/
  - /Game/Resources/
  - /Game/Scripts/
  - Move the scene to /Game/Scenes

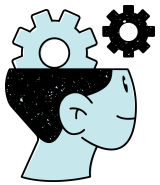




Exercise 2: Place a reference sphere at (0,0,0).

- This marks the expected game field center:





Exercise 3: Import the stadium model. Unpack the prefab and create your own version.

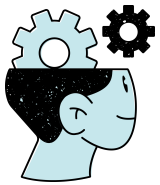
- Actions:

1) Import stadium model
   - Soccer Stadium Unity Asset Store
   - Move all the imported contents to the folder **/Game/Resources**

2) Load the scene **(Hayq Art/Scenes/Demo)** and clean the scene until you have only the stadium you need.

3) With the cleaned stadium, create custom prefab under **/Game/Resources** and name it **GameField**

4) Position stadium considering the previous sphere should be at the center



Exercise 4: Add invisible boundary walls using cubes.

- Actions:

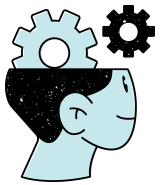
1) Create cubes around arena perimeter (create also a ceiling)

2) Disable MeshRenderer

3) Keep BoxCollider

4) Adjust positions to prevent leaving play area

5) Test it



Exercise 5: Creation of the **GameController** script and changing player's speed on joining.

- Actions:

1) Write the an empty script **GameController** in a new folder called **/Game/Scripts/Controller/**

2) Create an **Udon C# Program Asset** and rename it GameController

3) The GameController script should extend from (**UdonSharpBehaviour**)

4) Set the source script in the Udon C# Program Asset with GameController

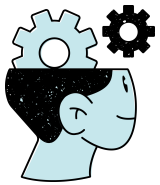
5) Create an empty gameObject in the scene and attach the script GameController.

6) There is a method inherited from (UdonSharpBehaviour) named (**OnPlayerJoined**). Overwritte it to display a debug message. (Debug.**Log**("On Player Joined");)

7) Find out how to change the speed of movement of the joined player if it's the local player (ask AI or search for it)

Code

```
public override void OnPlayerJoined(VRCPlayerApi player) {
         Debug.Log("On Player Joined");
          if (Networking.LocalPlayer == player)
          {
              player.SetWalkSpeed(6f);
              player.SetRunSpeed(8f);
          }
    }
```



Exercise 6: Import a skybox and set in **RenderSettings** the skybox for the game.

- Actions:

1) Import skybox

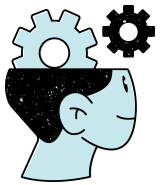
2) Create a member variable in **GameController** for the skybox material

3) Assign skybox through **RenderSettings** in Start method

4) Test scene visuals

Code

```
[SerializeField]
private Material sky;
void Start()
{
   RenderSettings.skybox = sky;
}
```



Exercise 7: Build & Test the arena in VRChat.

- Actions:

1) Use VRChat SDK → Build & Test

2) Enter world on desktop

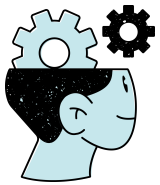
3) Optionally test on VR headset

`Code Checkpoint: GameField`

LESSON 02 COMPLETED

You now have:

- A fully built arena
- Boundaries that keep players inside
- A functional GameController
- A complete environment ready for gameplay logic



Self-Evaluation

It’s time to put what we’ve learned into practice! Here are 4 questions to check by yourself what you have learnt in this lesson.

- Question 1
- Question 2
- Question 3
- Question 4

Why is it important to create a clear folder structure in a Unity project?

- Because an organized project improves readability, maintenance, and teamwork
- Because Unity requires a specific folder structure to compile scripts
- Because VRChat will not upload worlds without a /Game/ folder

What is the main purpose of placing a reference object (like a sphere) at position (0, 0, 0)?

- To create a visible decoration for the arena
- To mark the center of the stadium for lighting calculations
- To understand spawn orientation and position relative to the world origin

Why are invisible boundary walls created using colliders without MeshRenderers?

- To reduce the number of objects in the scene
- To make the arena visually larger
- To restrict player movement without affecting visual appearance

What is the purpose of the basic GameController script introduced in this lesson?

- To manage ball physics and collisions
- To apply global logic affecting players, such as movement speed
- To control scoring and game rules

Help us to improve

Did you understand how to structure folders and organize the project?

Write your answer here.

Send

Was building the arena (floor, walls, stadium) clear to you?

Write your answer here.

Send

Did you struggle with scale, colliders, or camera positioning?

Write your answer here.

Send

What part of the arena creation needs more explanation?

Write your answer here.

Send