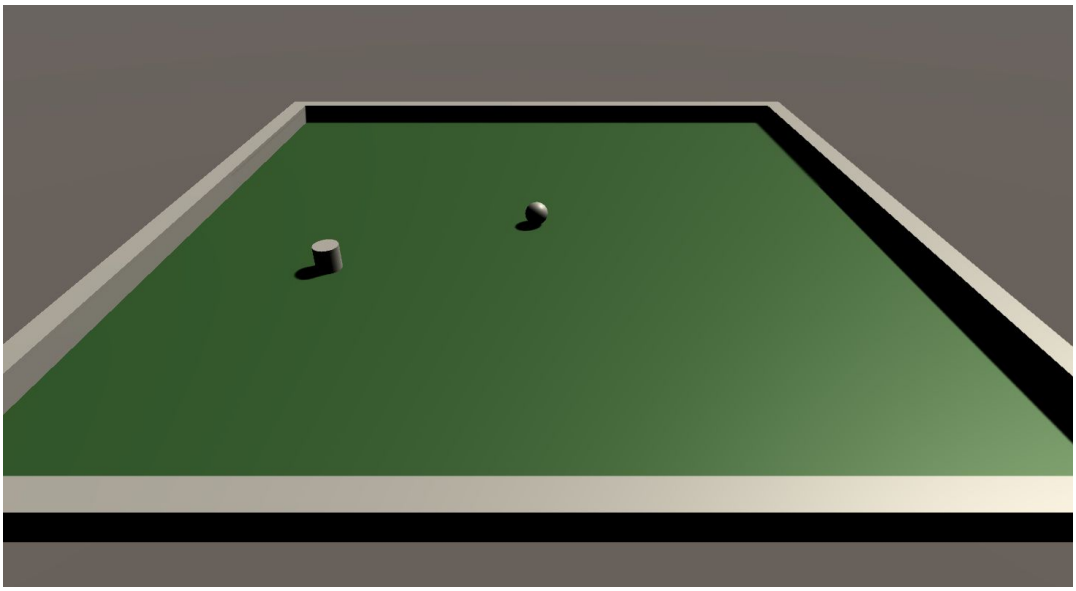
Exercise 0 - Welcome to Unity

In this exercise you will create a small game in which the player controls a sphere that collects cylinder-shaped “food” to gain points (so much fun!). The goal of this exercise is to learn about projects, scene navigation, scripting, and Unity in general.

In the TA slides you can find a video demonstrating how the final game should look & behave.

This exercise is **not** for submission but it is highly recommended to complete it in order to prepare for the rest of the course and future exercises.



Part 0 / Setup

1. Download the file *EX0.zip* from the course Moodle website and unzip it somewhere on your computer.
2. In Unity Hub, go to *Projects* and click the *Add* button on the top right. Select the folder EX0 that you have downloaded.
3. Open the project. Once Unity is open, double click the MainScene in the project assets view, and you should see an empty playing field appear in the Scene View.
4. Try moving around the scene, inspect the scene hierarchy and get to know the user interface.

Part 1 / Player GameObject

1. Add a sphere GameObject to the scene hierarchy (Right click in the hierarchy view → 3D Object → Sphere). This will be our player - rename it to *Player*.

2. Select the Player GameObject in the scene hierarchy - It will then appear in the Unity Inspector on the right side of the screen. Using the transform component, position the sphere at (0, 1, 0)

3. Add a `Rigidbody` component to the sphere. This will allow our sphere to use Unity's physics engine to interact with the world and other GameObjects

4. Add the PlayerController Script component to the Player GameObject and open the script

Remider: To add components to a GameObject, use the "Add Component" button in the inspector. Alternatively, you can just drag a component (e.g. a script) onto the GameObject or the inspector.

Part 2 / Player Script

1. In the `Start` method, use `GetComponent` to initialize the `Rigidbody` class property

2. Allow the sphere to be moved using the arrow keys: Use the `AddForce` function of the Rigidbody with the appropriate direction vector. The up arrow should move the ball in the `Vector3.forward` direction, right arrow in the `Vector3.right` direction, etc.

   Multiply the direction by the `movementForce` property that will control the movement power. Don't forget to multiply everything by `DeltaTime`!

   To check for user input you may use `Input.GetKey` with the correct KeyCode.

3. In the Unity UI, click ▸ Play to enter play mode. Try moving around the field and changing the `movementForce` to see how it affects the player.

   Click ▸ Play once more to exit play mode. Note that when exiting play mode the `movementForce` parameter returns to its original value.

Part 3 / Game Controller

1. In the scene hierarchy, create an empty GameObject (Right click in the hierarchy view → Create Empty) and rename it *GameController*.

2. Add the GameController script component to the GameObject and open the script.

3. We want the food object to be drawn as a cylinder in the game. In the `Start` method, initialize the food GameObject using `CreatePrimitive`.

4. Implement the `SpawnFood` method. The method needs to position our food at a random location inside the field bounds. Note that the field is positioned on the XZ plane with a width and height of `FIELD_SIZE`.

   To get a random number, you may use the `Random.Range` function.

5. If we want to access the Player GameObject from within our GameController script, we need to link them in the Unity UI.

   The class property playerObject is public, meaning we can see and edit it from the inspector. Drag the Player GameObject from the scene hierarchy to the public field. Now we can access the player properties and functions using the playerObject reference.

Drag Here ↑

6. In each frame, we need to check if the player has collected the food. If the distance between the food and the player is less than `COLLISION_THRESHOLD`, call `SpawnFood` to move the food elsewhere in the field.

7. Add a score property to the GameController class and initialize it to 0. Every time the player collects food, increment the score and print it to the console.

Part 4 / Camera Tracking

1. In the GameController script, add a public GameObject property called cameraObject, which we will use to reference the camera. Connect it from the inspector, like we did with the playerObject.

2. Make the camera position follow the player around.

   Hint: first save the initial offset between the camera position and player position.