Current Trends in Gaming

Module 2

Make a clone: Osmos

In this exercise you will implement a simple version of Osmos using the Unity game engine. For reference, an executable of the completed version of this exercise will be uploaded to the Canvas. Asset files such as images and sound files are also bundled with this zip file.

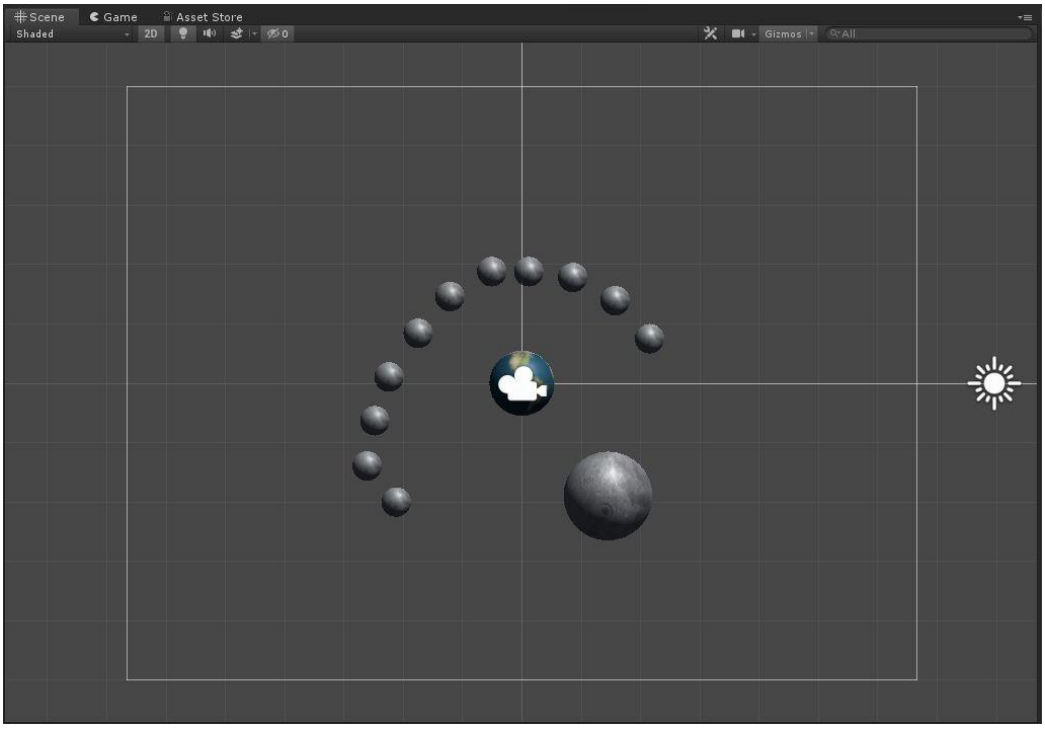
Task 1. As with Pachinko, start a new project in Unity with the 2D template and add a light source.

Task 2. Create a player Sphere in the centre of the scene. Add to it a Rigidbody, deactivate "Use Gravity" and set Drag to 2.

Create a script for the player sphere. The sphere should move according to which arrow keys are pressed. The movement should come from forces being applied to the sphere's Rigidbody. When the sphere moves it should slightly lose mass/volume. You can model the mass/volume by using the sphere's Scale.

Task 3a. Create other spheres and place them into the scene.

Aim for something similar to this setup (the textures will come in a later task):



Task 3b. If a player sphere comes into contact with one of the other spheres, the bigger one of the two should start taking mass/volume from the other until the smaller one has a 2D area smaller than 0.01, in which case it is removed from the scene.

Task 4. Add sounds.

Add the “blop.mp3” and “ting.mp3” files to the project.

1. When a sphere is removed from the scene play the “Blop” sound.
2. When all spheres have been removed, play the “Ting” sound. Make sure to have exactly one Audio Source in the scene.

Task 5. Add UI elements.

1. At the top of the screen, display the current mass/volume of the player sphere.
2. When all spheres have been removed, display the text “YOU WIN!” in the middle of the screen.
3. If the player reaches a smaller 2D area than 0.01 make the player sphere disappear and display “GAME OVER” in the middle of the screen.

Task 6. Add textures to the GameObjects.

Add the “earth.jpg” and “moon.jpg” files to the project. Apply the earth texture to the player sphere and the moon texture to the other spheres.
To apply a texture to an object, simply drag it from the Assets window onto the object in the scene.

Task 7. Add background.

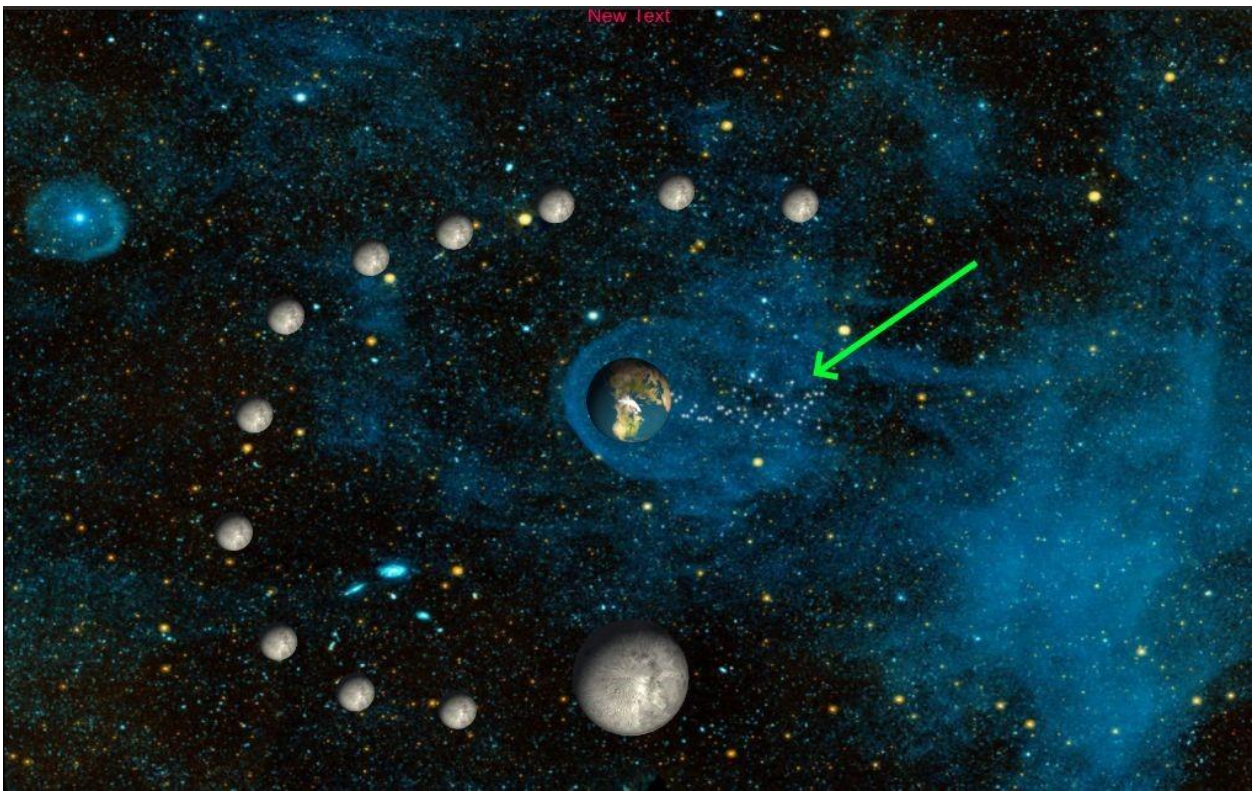
Create a new Camera and a Quad.

1. Make the camera orthographic and move it to position (30, 0, -10) (far out of view of the other camera).
2. Make the quad a child of the camera and position it in the middle of the camera view, then scale the quad to fill the view.
3. In order to blend the image of the original Main Camera and the new camera, and to make sure the new camera is rendered first, set the Main Camera to have Clear Flags “Depth only” and Depth 1 and the new camera to have Depth 0.
4. Add the “space.jpg” image file to the project and drag it from the Assets window onto the Quad in the scene. This will automatically create a new Material for the Quad. Change the Material’s Shader to Unlit/Texture.

Task 8. Add sphere rotation.

Make it so that the player sphere and the other spheres continuously rotate slowly about the z axis around their own centres. This will create a deep immersive feeling in the game!

Task 9. Add particle exhaust behind the Player sphere opposite its direction of motion.



1. Create a Particle System (Effect) GameObject and position it at the centre of the Player sphere.
2. Adjust the shape and scale of the emitting particle system so that it looks decently like an exhaust.
3. Set the particle system's Simulation Space to "World" to make the position of already emitted particles unaffected by the particle system's transformation.
4. Create an Exhaust script that takes in the Player GameObject as a public member variable input.
   The script should make sure that the particle system's translation matches the translation of the Player sphere, the direction of the emission (i.e., particle system's rotation) should be opposite the Player sphere's velocity direction, while the emission rate should be based on the Player sphere's speed.