

Unity Entry Project

CMPE 485

1 Introduction

This project aims to teach the basic features and workflow of the Unity engine. By the end of the project, the following topics will have been covered:

- Object creation
- Object translation, rotation, scaling
- Rigidbodies and physics
- Physics materials
- Materials and Textures
- Scripting and Inputs
- Lighting and Camera details
- Animation
- Sound

2 Steps of the project

2.1 Create a platform

A platform GameObject should be created. This platform should not be affected by forces, collisions, or gravity. It should possess a RigidBody. Optionally, it should be surrounded by walls.

2.2 Creating moveable objects

Like the platform above, multiple other GameObjects should be created. These objects should be affected by forces, collisions, and gravity. These should also have RigidBodies.

2.3 Adding physic materials

Physic Materials should be added to the movable objects to see how they behave on surfaces with different values. To test these out, immovable slopes could be added. Keep in mind that Unity physics are not realistic, and objects like spheres will only consider a single contact point, and physics materials may not behave as you would expect them to. Using cubes is preferable.

2.4 Handling Collision

Collision between objects should be tested by placing the moveable objects in places where they can naturally be affected by gravity. The immovable platforms should stay put but affect the dropping objects regardless.

2.5 Creating scripts

A sample script should be created in the project folder. Preferably, it should have its own Scripts folder. The script should derive from MonoBehaviour. The script then should be added to one of the moveable objects.

2.6 Applying forces to RigidBodies

The script should take the RigidBody component of the object and add a certain amount of force to it every frame. The script can acquire the RigidBody via Unity's GUI by dragging and dropping, outside of the script itself.

2.7 Adding textures

Materials should be created using images, which can then be assigned to the Mesh Renderer of any 3D object. The image can be added to any of the fields of the material (Albedo, Metallic, etc.), proving different results.

2.8 Prefabs

Prefabs are an easy way to save a certain object that you created so that it can be used and spawned as necessary. A prefab from any of the movable objects should be created and must have a script that can be used to spawn new objects by pressing a button on the keyboard.

2.9 Adding lighting

Various sources of lighting should be added to the scene. Be mindful that lighting systems work off of the GPU.

2.10 Modifying the Camera

The camera should be modified to show a better view of the scene. Its position, rotation, scale, Field of View, and more can all be adjusted inside the scene.

2.11 Making objects respond to inputs

Games require a player to function. By adding a way to check for inputs inside a script, we can control an object. For this exercise, the object should respond to inputs by adding force in the direction the inputs were received from, preferably by using WASD or arrow keys.

2.12 Creating an animation

Make a simple animation that scales a cube up and down.

Optional: Create additional animations and establish an animation flow using an animation controller.

2.13 Play a sound

Create a sound source and play an MP3 sound.

Optional: Implement a functionality where pressing the 'm' key stops the sound, and pressing 'm' again resumes the sound clip.

2.14 (Optional) Making the camera follow the player

Games need to have the camera focus on the play area so that the player does not miss out on any action. The position of the camera depends on the type of game (first-person shooters vs. 2D platformers). The camera in this case should follow the "player object" that was defined earlier. Even though this can be done via a script that is applied to the camera, there is a much simpler way to accomplish this.

IMPORTANT:

Please do not push the entire project to GitHub at once. Commit the steps in the project step by step.

Please do not forget to add .gitignore for Unity to your project.

Upload a brief video demonstrating the project's steps to Moodle.