**Assignment # 2**



**Fall 2024**

**CSE-411 Intro to Game Development**

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

**Engr. Abdullah Hamid**

Date:

20**th November 2024**

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Task:**

As discussed in class, create a new scene with a plane the camera should be top down over the plane. You must instantiate enemy cubes on the plane on random location (they must not appear out of the plane area). Also they must destroy themselves after 2 seconds.

**Solution 1:**

**Theory:**

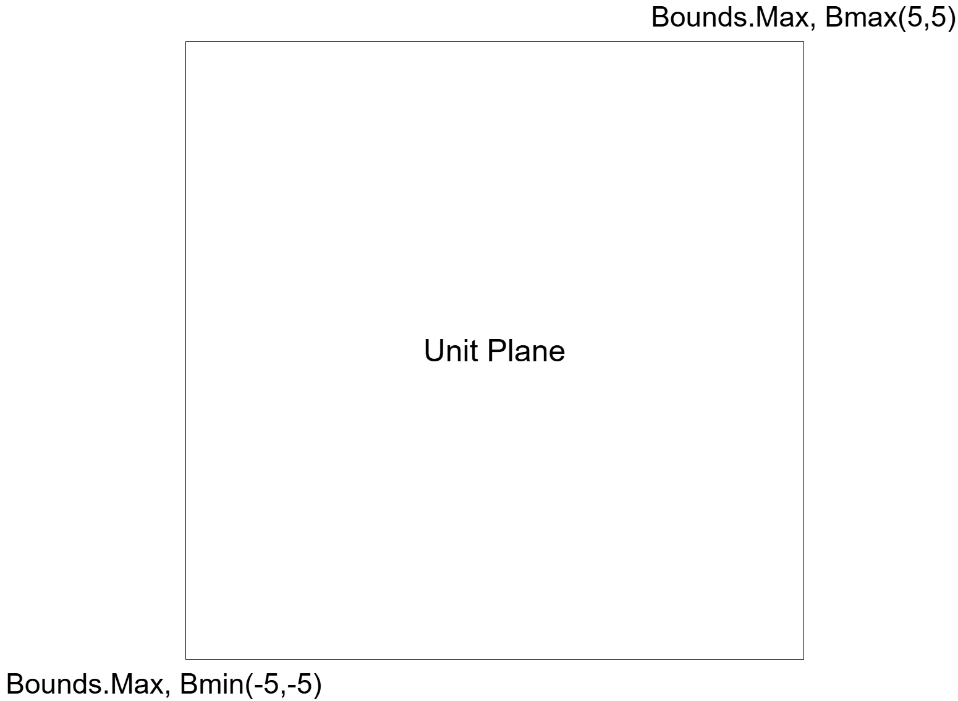
****

Figure 1: Bounds of Unit Plane in Unity

In figure 1, we can see a unit plane in Unity’s 3D world space. To get the total area of plane, we can get the bounds using the MeshRenderer component of plane. Now to spawn enemy cube inside this plane’s area, we need to find a random point inside this plane area using the minimum and maximum vector of bounds. For safety reasons, I have also added an offset of 0.5 to avoid instantiating the enemy cube outside the plane. This solution gets continuous random positions of the plane.

**Code:**

**EnemySpawner class**

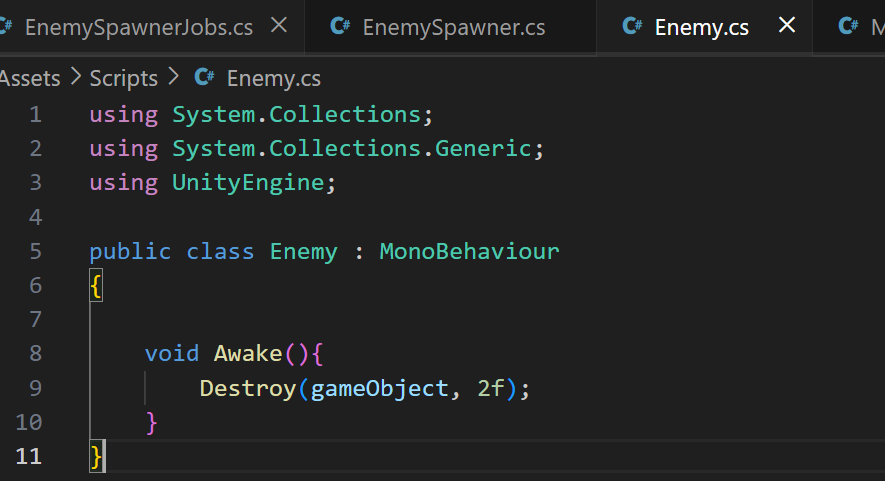
**A screenshot of a computer program

Description automatically generated**

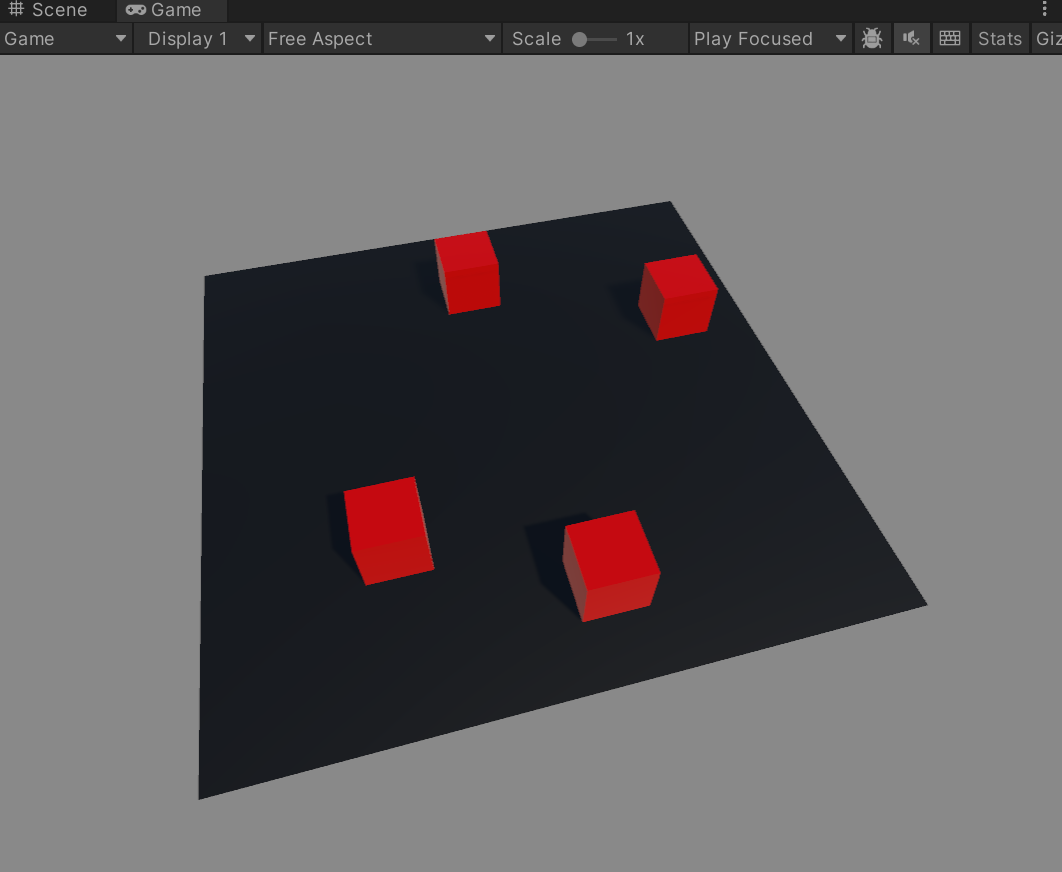
**A screen shot of a computer code

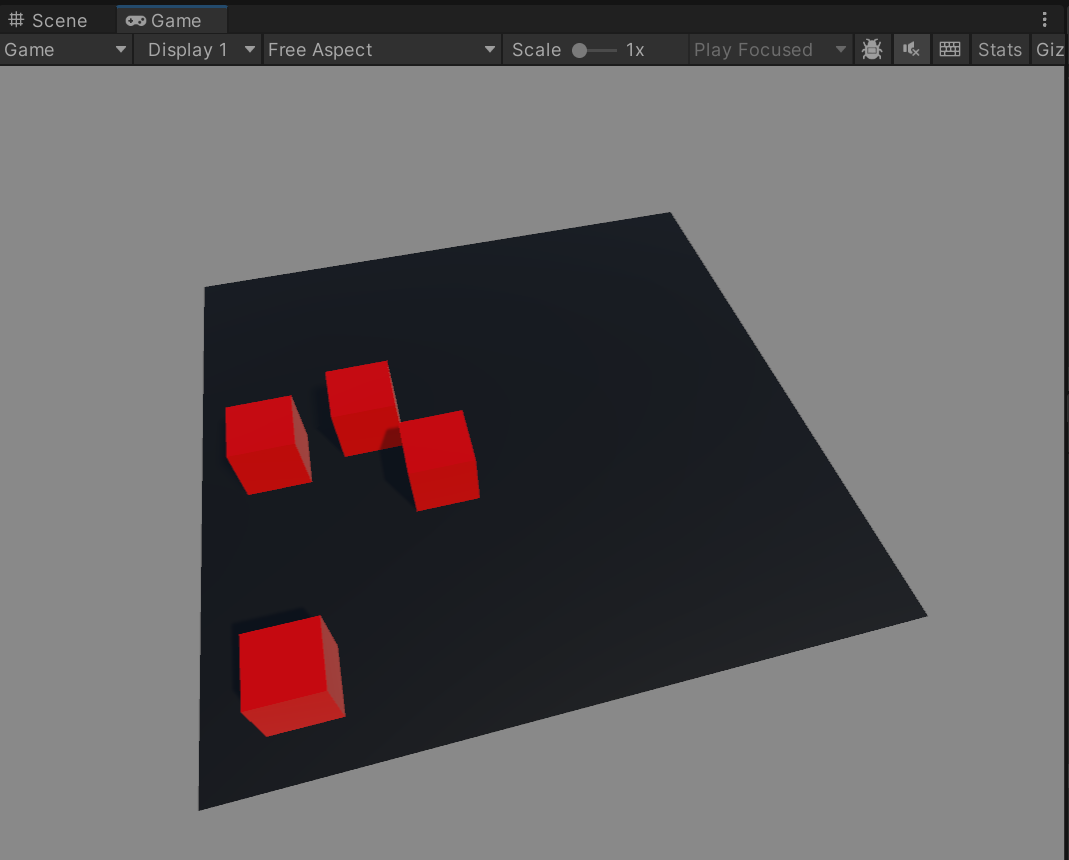
Description automatically generated**

**Enemy class**

****

**Output:**

****

****

**Solution 2:**

**Theory:**

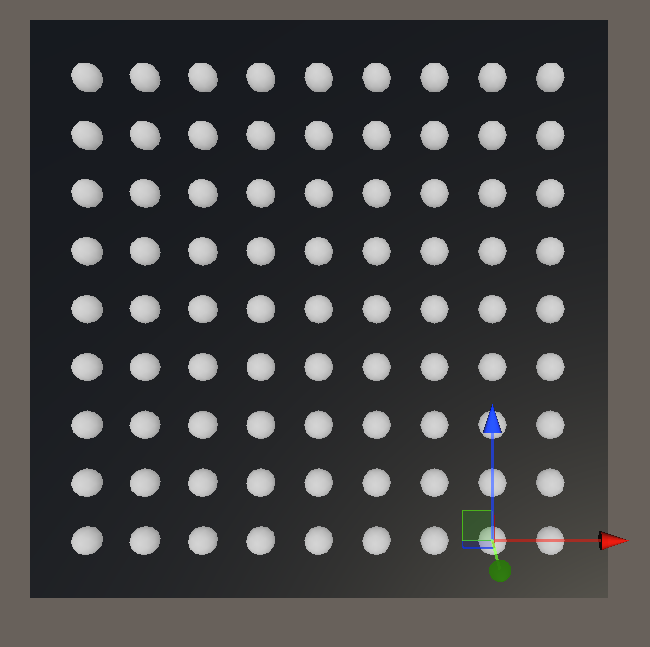


Figure 2: Discrete Points inside Unit Plane in Unity

A grid of white dots

Description automatically generated

Figure 3: Discrete Cells inside Unit Plane in Unity

In figure 2, the concept is almost same but, in this solution, I’ve discretized the random points inside the plane. I have first saved all the points in a vector2 list and then selected a random point from this list for instantiating the enemy cube. These discrete points inside the plane bounds/area can be referred to as cells. For visualization, I have used the OnDrawGizmos function available in Unity for visual debugging. In figure 3, I have also draw the cells around each point.

**Code:**

**EnemySpawner class**

**A screen shot of a computer code

Description automatically generated**

**A screen shot of a computer code

Description automatically generated**

**Enemy class**

This class remains the same as before.

**Output:**

**A screenshot of a game

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**