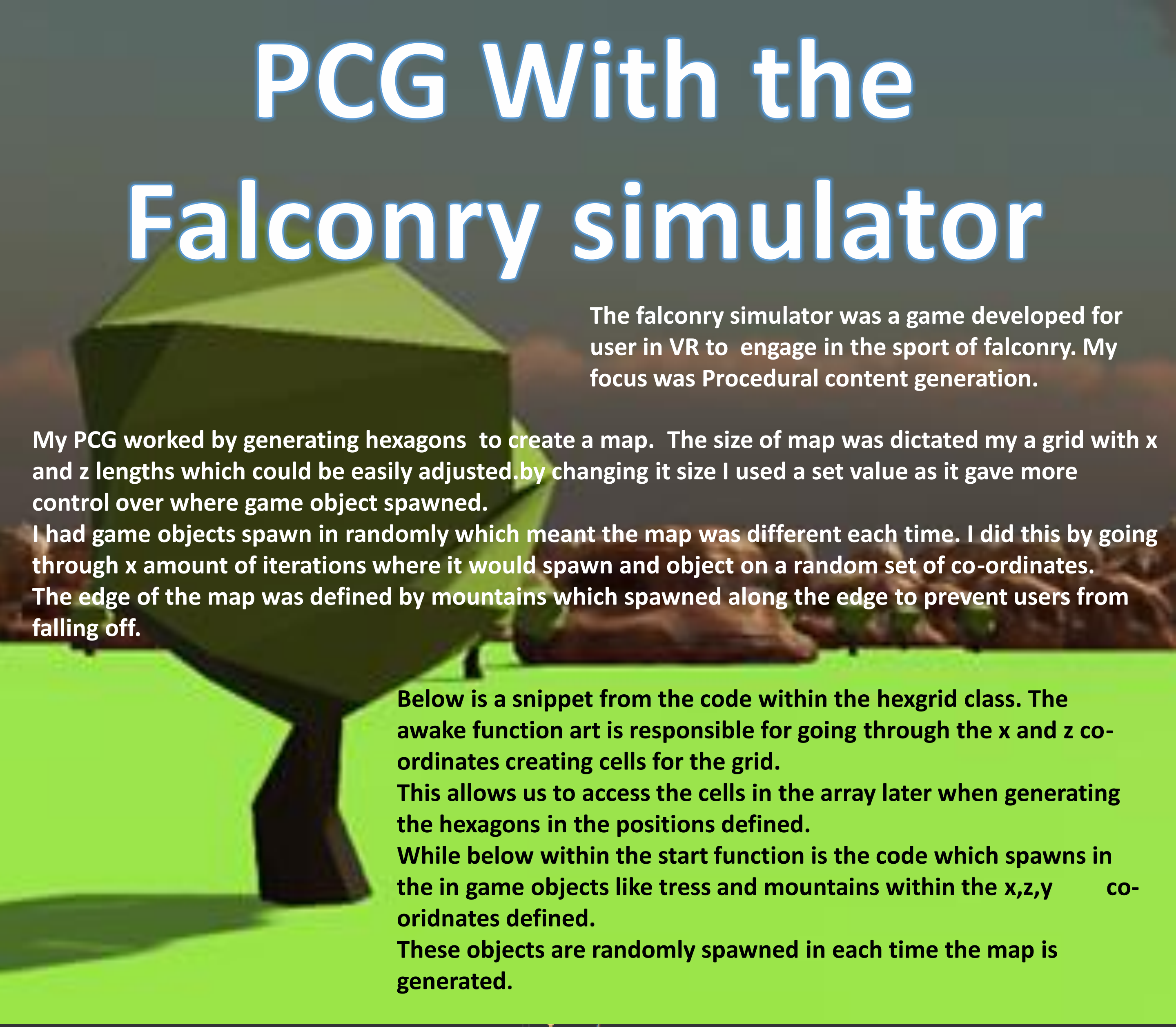
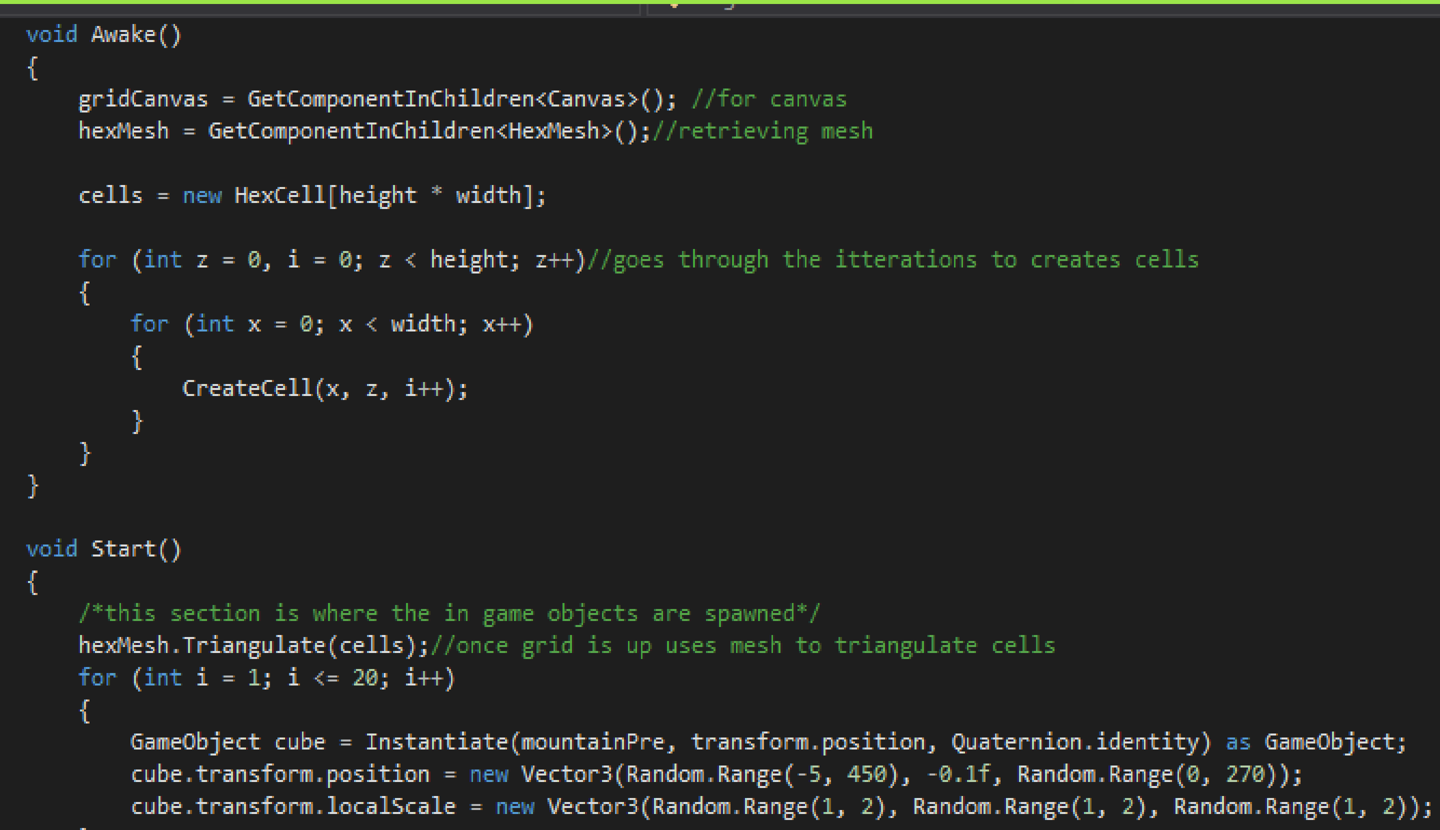
PCG With the Falconry simulator



The falconry simulator was a game developed for user in VR to engage in the sport of falconry. My focus was Procedural content generation.

My PCG worked by generating hexagons to create a map. The size of map was dictated my a grid with x and z lengths which could be easily adjusted.by changing it size I used a set value as it gave more control over where game object spawned.
I had game objects spawn in randomly which meant the map was different each time. I did this by going through x amount of iterations where it would spawn and object on a random set of co-ordinates.
The edge of the map was defined by mountains which spawned along the edge to prevent users from falling off.

Below is a snippet from the code within the hexgrid class. The awake function art is responsible for going through the x and z co-ordinates creating cells for the grid.
This allows us to access the cells in the array later when generating the hexagons in the positions defined.
While below within the start function is the code which spawns in the in game objects like tress and mountains within the x,z,y co-oridnates defined.
These objects are randomly spawned in each time the map is generated.



```
void Awake()
{
    gridCanvas = GetComponentInChildren<Canvas>(); //for canvas
    hexMesh = GetComponentInChildren<HexMesh>();//retrieving mesh

    cells = new HexCell[height * width];

    for (int z = 0, i = 0; z < height; z++)//goes through the itterations to creates cells
    {
        for (int x = 0; x < width; x++)
        {
            CreateCell(x, z, i++);
        }
    }
}

void Start()
{
    /*this section is where the in game objects are spawned*/
    hexMesh.Triangulate(cells);//once grid is up uses mesh to triangulate cells
    for (int i = 1; i <= 20; i++)
    {
        GameObject cube = Instantiate(mountainPre, transform.position, Quaternion.identity) as GameObject;
        cube.transform.position = new Vector3(Random.Range(-5, 450), -0.1f, Random.Range(0, 270));
        cube.transform.localScale = new Vector3(Random.Range(1, 2), Random.Range(1, 2), Random.Range(1, 2));
```