## Sprint 4 – Implementing the Mesh Generator class to display the heightmap 27/02/2019 – 08/03/2019

### Abstract

The goal of this sprint is to create a class to generate a basic mesh that can be generated during run time and in the unity editor. When the class is created the goal is to generate the noise map with Perlin Noise and add the values to the Y component of the vertex vector.

### Research / Implementation

I already had previously knowledge in how to create meshes in DirectX, and this uses the same principle.

#### Mesh Class

##### Implementation

The first thing when creating the Mesh Class, the author created a simple sketch and an UML design for the class [Figure 1] [Figure 2].

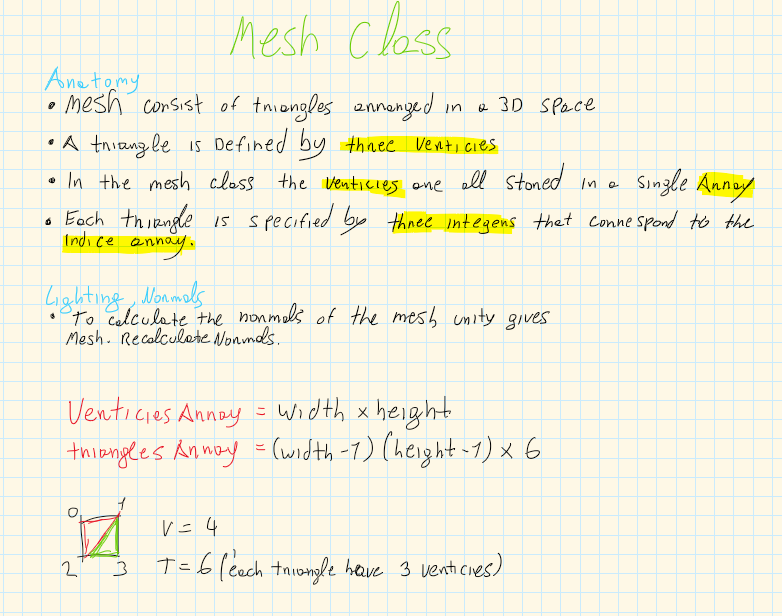


Figure 1 - Mesh Class Sketch

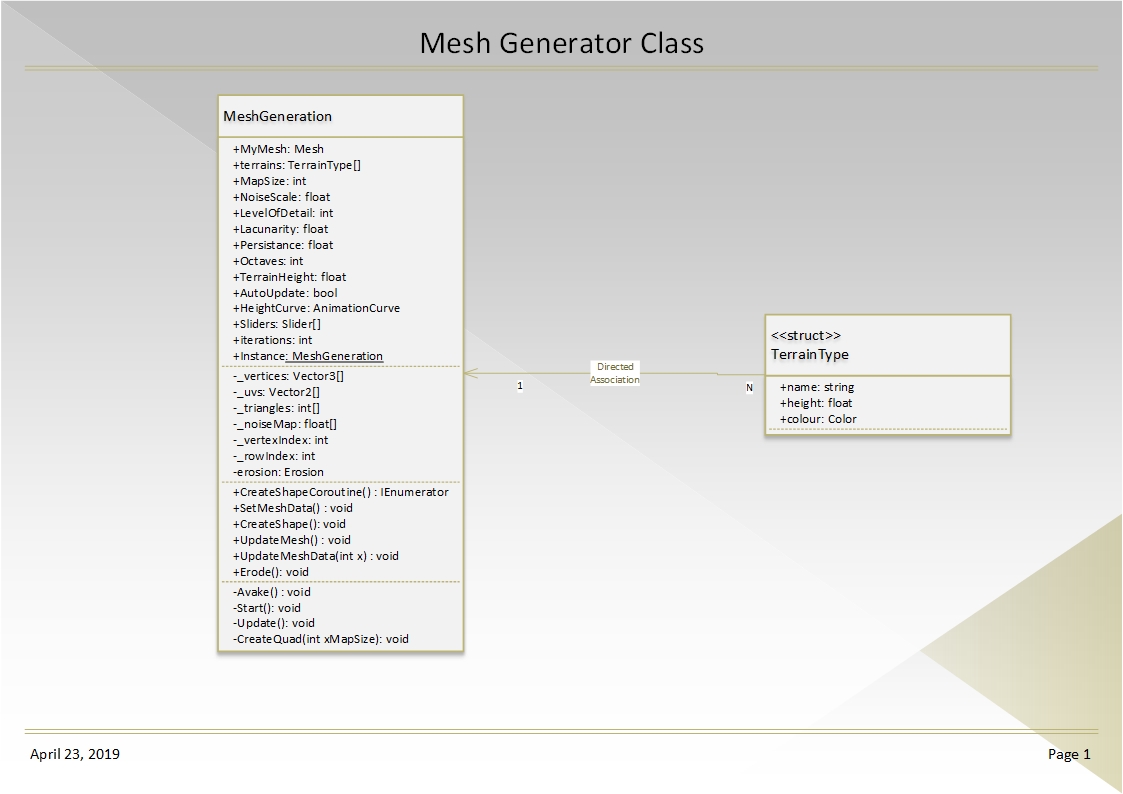


Figure 2 - Mesh Generator Class Design

The second thing to make sure is that the empty game object has attached the following two components: Mesh Filter and Mesh Renderer [Figure 3].

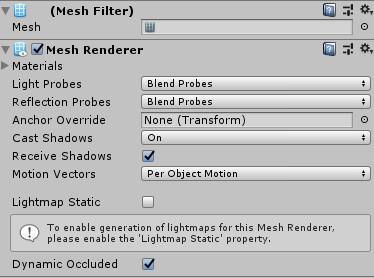


Figure 3 - Requirement of these two components

With the sketch in mind [Figure 1] three initial method were created for the class, SetMeshData(), CreateShape() and UpdateMesh().

###### Set Mesh Data method

This method creates a new mesh, sets the Mesh Filter that is attached to the Empty Game Object to be equals the new mesh and makes sure that the main texture in the Mesh Renderer is set to null. Then it initializes the vertices, triangles and UVs. The initial sketch that the author did to initialize the vertices [Figure 1] was wrong, the fixed result can be seen in [Figure 4], the reason why the improved version has + 1 derives from the fact that the author wanted to have 255 quads when the Map Size was equals to 255.



Figure 4 - Vertices, triangles and UVs initialization

###### Create Shape method

This method is responsible for going through each vertex and UV and initialize them with a defined value and passing the height of the noise map to the vertex [Figure 5].

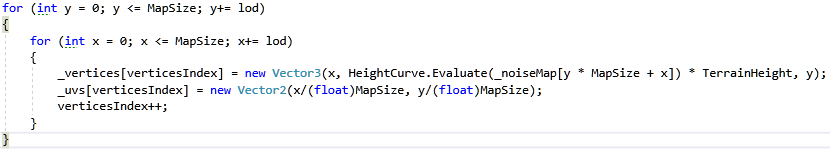


Figure 5 - Vertex and UVs initialization

The other functionality this method is responsible is for creating the triangles for the mesh [Figure 6].

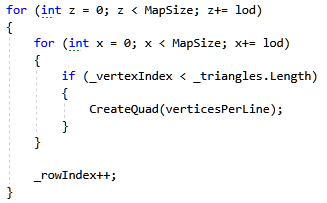


Figure 6 - Triangles generation

###### Animation Curve

To improve the noise map output and create a more realistic terrain the author implemented an animation curve from unity. The animation curve limits the height map range [Figure 8]. In other words, an animation curve in unity has multiple keys [Figure 7] in were the user can control the points in what that curve passes through (Unity Technologies 2018).

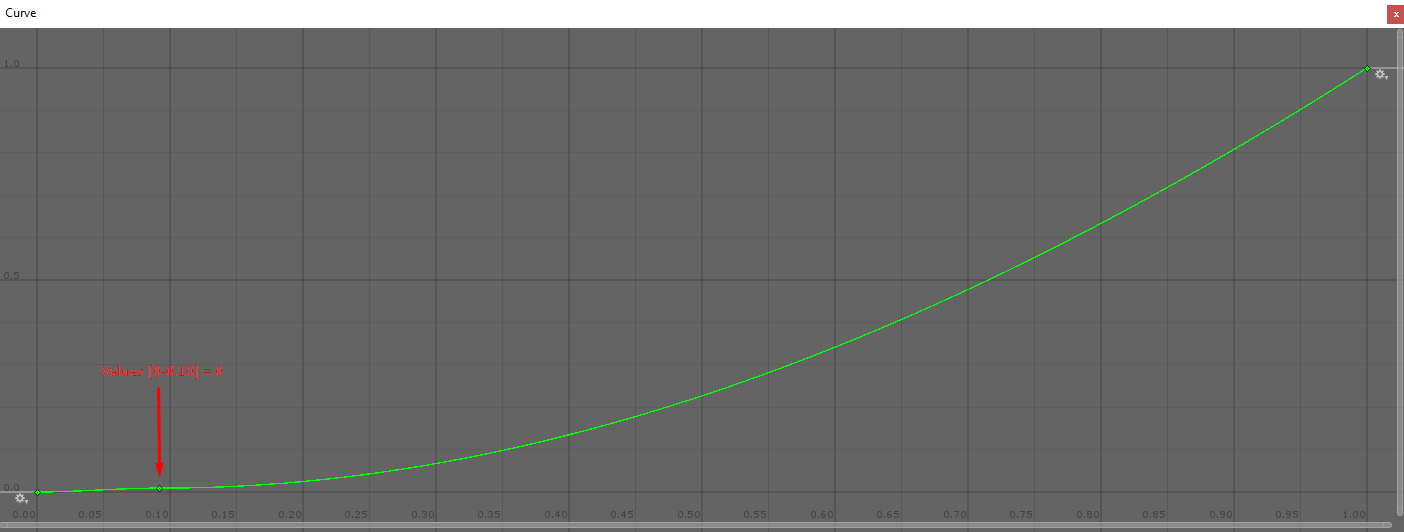


Figure 7 - Animation Curve Graph



Figure 8 - Animation Curve

#### Problems and Solutions

During the implementation of the class the author struggled with different problems. The first problem was found on the initialization of the vertices, for example; The vector3 named vertices had the size of . Let’s imagine for a moment that the Map Width and Map Height have a value of 10, creating a total size of 100 for the vector. The only problem is that the author wanted to draw 10 quads per row and only 9 were created. The problem was that the last vertex in the rightest corner was present in the other side of the map creating these strange connections [Figure 9].

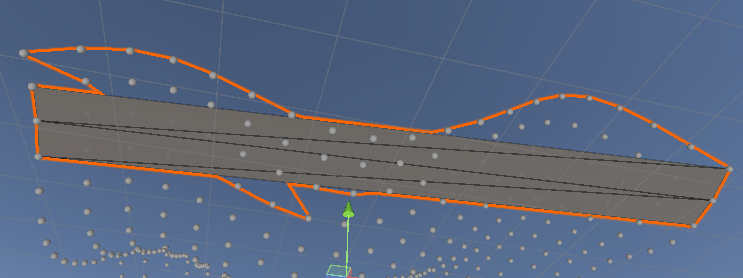


Figure 9 - Generation Mesh Problem

The solution for this was to create the vertices with the size of [Figure 10]

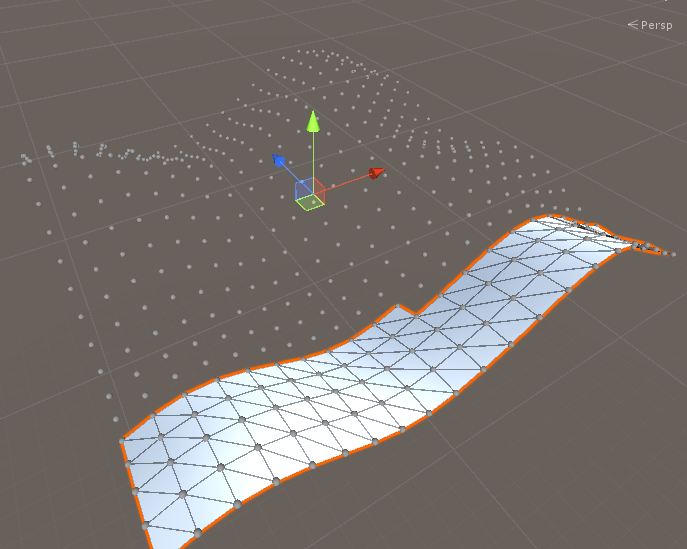


Figure 10 - Mesh Terrain Generation

The second problem was in how the terrain was getting renderer, to fix this problem the author found that limiting the persistence level between 0 and 1 and increasing the lacunarity to 2.6 fixed the problem [Figure 11].

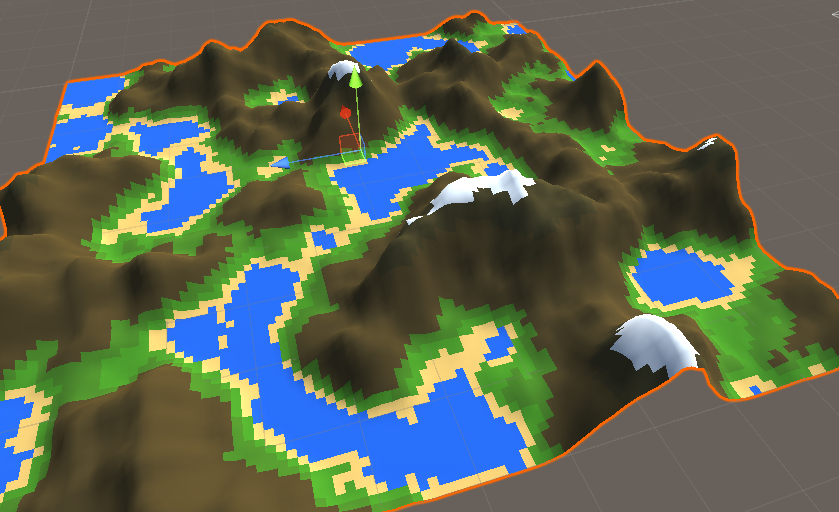
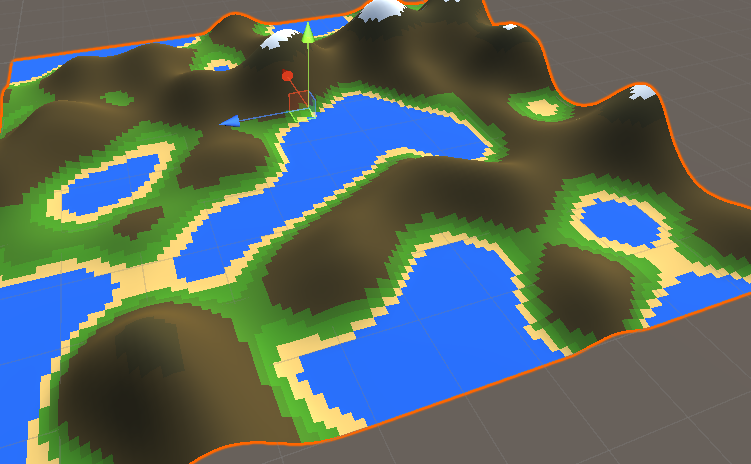


Figure 11 - Terrain Generation with low level of lacunarity | Terrain Generation with high level of lacunarity

### Sprint Review

The sprint was an overall success, in only 23 hours the author was able to have a terrain procedural generated. At this moment all the sprints are going ahead of time and no major problem was encountered so far. In this sprint the author not only learned how to effectively create a Mesh Class and how to manipulate the creation of a mesh by code but also how to use the animation curve to limit the values from the noise map.

### WBS

1. Research (50%) (12 hours)
2. Create a Mesh Class (40%) (9 hours)
3. Generate the Noise Map to add the height for each vertex (10%) (2 hours)

### Bibliography

ABHINAV A.K.A DEMKEYS, 2017. *Mathf.PerlinNoise | Smooth Random Values | Unity Quick Tutorial*[viewed 15/03/ 2019]. Available from: <https://www.youtube.com/watch?v=gdSFs0PeBNQ>

BRACKEYS, 2018. *MESH GENERATION in Unity - Basics*[viewed 15/03/ 2019]. Available from: <https://www.youtube.com/watch?v=eJEpeUH1EMg>

BRACKEYS, 2018. *PROCEDURAL TERRAIN in Unity! - Mesh Generation*[viewed 15/03/ 2019]. Available from: <https://youtu.be/64NblGkAabk>

FLICK, J., 2018. *Procedural Grid*[viewed 15/03/ 2019]. Available from: <https://catlikecoding.com/unity/tutorials/procedural-grid/>

UNITY TECHNOLOGIES, 2018. *Using Animation Curves*[viewed 12/03/ 2019]. Available from: <https://docs.unity3d.com/Manual/animeditor-AnimationCurves.html>

UNITY TECHNOLOGIES, 2019. *Procedural Mesh Geometry*[viewed 15/03/ 2019]. Available from: <https://docs.unity3d.com/Manual/Example-CreatingaBillboardPlane.html>

### References

UNITY TECHNOLOGIES, 2018. Using Animation Curves [viewed 12/03/ 2019]. Available from: <https://docs.unity3d.com/Manual/animeditor-AnimationCurves.html>