Read Me

This is the Space Graphics Toolkit (SGT) **Terrain** feature. This contains components allowing you to render planet and star surfaces with much higher detail than you can achieve with a single sphere mesh. This is done using LOD, where the mesh detail increases as you approach the surface.

NOTE: This feature is designed to make high detailed surface meshes with some procedural generation, it's not designed to generate realistic 1:1 scale planets like Earth or even the Moon.

Making a terrain

Begin by right clicking in your Hierarchy and selecting: **Space Graphics Toolkit → Terrain**
Or from the menu bar selecting: **GameObject → Space Graphics Toolkit → Terrain**

Your scene should now contain a new selected GameObject called "**Terrain**" with the **SgtTerrain** component.

You should then set the **SgtTerrain.MaterialSgtTerrain** to the material of your choosing.

NOTE: For performance reasons terrains use cube mapped UV data, not equirectangular (cylindrical) projection. So you cannot use the normal materials you put on sphere planets. Follow the "**Creating Cube Textures**" documentation to learn how to convert them and use the **SgtTerrainCubeMaterials** component to assign them.

Adding LOD

By default a terrain won't have any LOD. To add some, you need to fill the **SgtTerrain.Distances** list.

If you click the "**Add Distance**" button, then either a distance of 1 will be added, or half the value of the last distance in the list will be added.

The distance values are are local space, so if your terrain has a scale of 1,1,1, and it is in the root of your Hierarchy, then a distance of 100 means a distance of 100 meters is required for any terrain surfaces with the current LOD level to split into 4 smaller children. These children then use the next LOD distance to calculate if they should split or not.

Adding Height

By default the terrain is just a round sphere. To add height, you can add the **SgtTerrainHeightmap** component. You can then set the **SgtTerrainHeightmap.Heightmap** texture to one that stores the terrain height.

NOTE: This texture must be marked as readable in the texture import settings.
NOTE: This texture should be use equirectangular (cylindrical) projection.

Adding Procedural Height

The **SgtTerrainSimplex** and **SgtTerrainRidgedSimplex** components can be added and stacked with different settings to generate procedural height for your terrains.

Spawning Objects

If you want to procedurally spawn objects like trees on your terrain then you can add the **SgtTerrainSpawner** component.

The **SgtTerrainSpawner**.**Prefabs** list can contain any prefab with the **SgtTerrainObject** component attached. This component is used to manage object pooling, procedural scaling, and other features.

The **SgtTerrainSpawner.Depth** setting allows you to set which LOD depth you want the objects to appear on. If you set this to 1 then your terrain surface must split at least once for the objects to appear.

The **SgtTerrainSpawner.Probability** setting is used to procedurally choose if an object is spawned on the current terrain surface if it's on the correct depth. This probability is checked multiple times until it either fails or it reaches the **SgtTerrainSpawner.SpawnCountMax** setting.

You can also set the **SgtTerrainSpawner.HeightMin/Max** settings to control the heights that objects can be spawned on. This is in local space and corresponds to the **SgtTerrain.Radius** setting + any height modifications.