// ProceduralTerrain

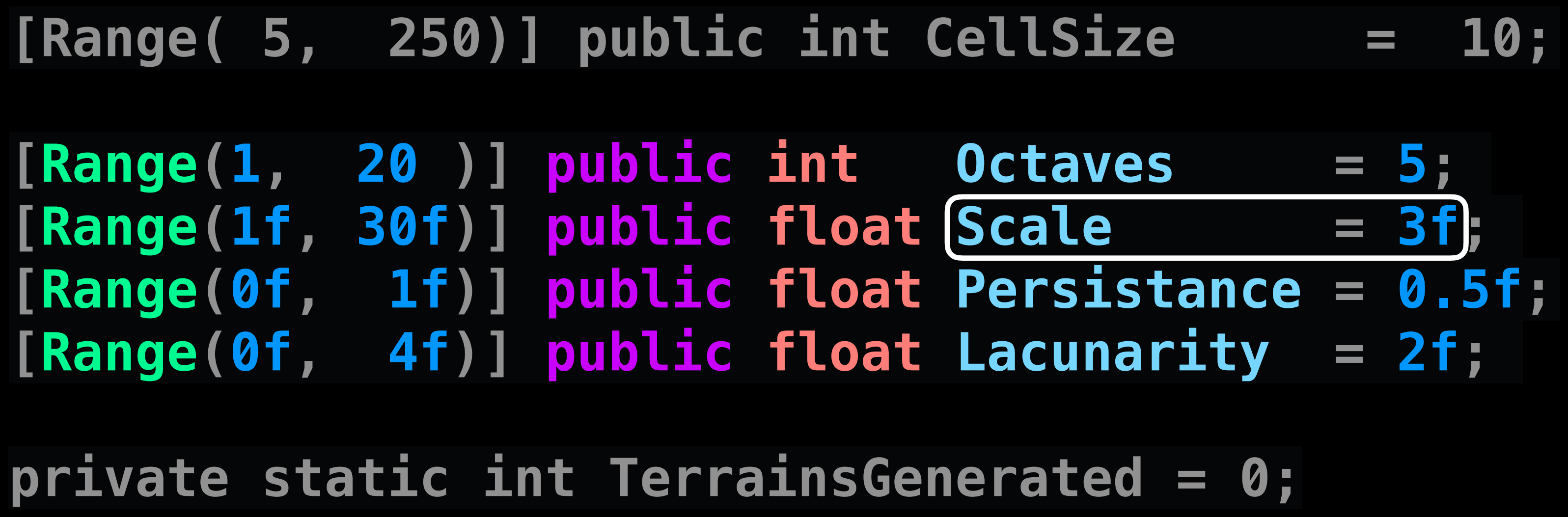
```
[Range( 5,  250)] public int CellSize      =  10;

[Range(1,  20 )] public int   Octaves     = 5;
[Range(1f, 30f)] public float Scale       = 3f;
[Range(0f,  1f)] public float Persistance = 0.5f;
[Range(0f,  4f)] public float Lacunarity  = 2f;

private static int TerrainsGenerated = 0;
```

Another way to say it:
larger values mean we're more
"zoomed in" on the Perlin Noise

// ProceduralTerrain



```
[Range( 5,  250)] public int CellSize      =  10;

[Range(1,  20 )] public int   Octaves     = 5;
[Range(1f, 30f)] public float Scale       = 3f;
[Range(0f,  1f)] public float Persistance = 0.5f;
[Range(0f,  4f)] public float Lacunarity  = 2f;

private static int TerrainsGenerated = 0;
```

This probably sounds really weird and confusing

It will make more sense when we step through the height equations soon