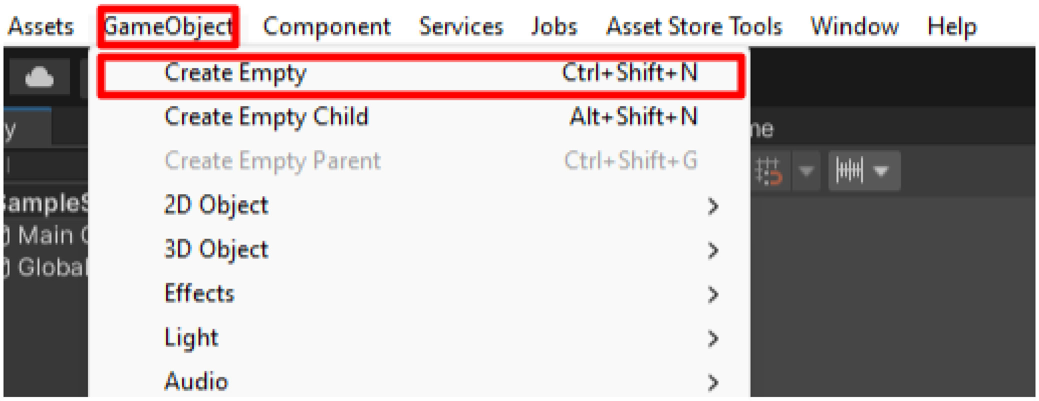
DOCUMENTATION

Welcome to *Easy World Generation 2D, a fast and easy-to-use* tool that allows you to create custom 2D worlds in just a couple of seconds!

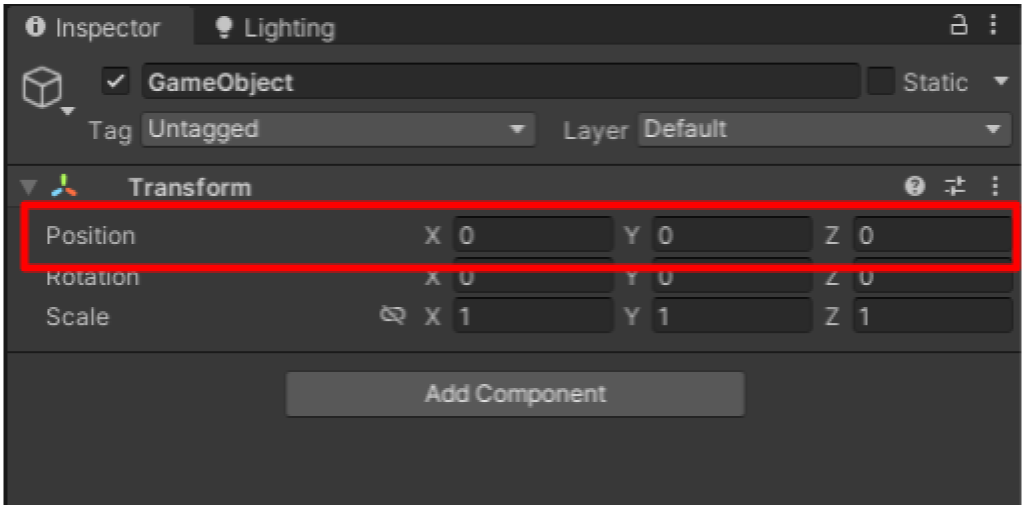
SETUP

Once you have the *Easy World Generation 2D* package installed in your Unity project, there are just a couple of steps you need to do before it gets up and running.

1) Create an empty GameObject (GameObject > Create Empty)



2) Change the Transform to (0,0,0), or any other position you like



GameObject

Tag Untagged

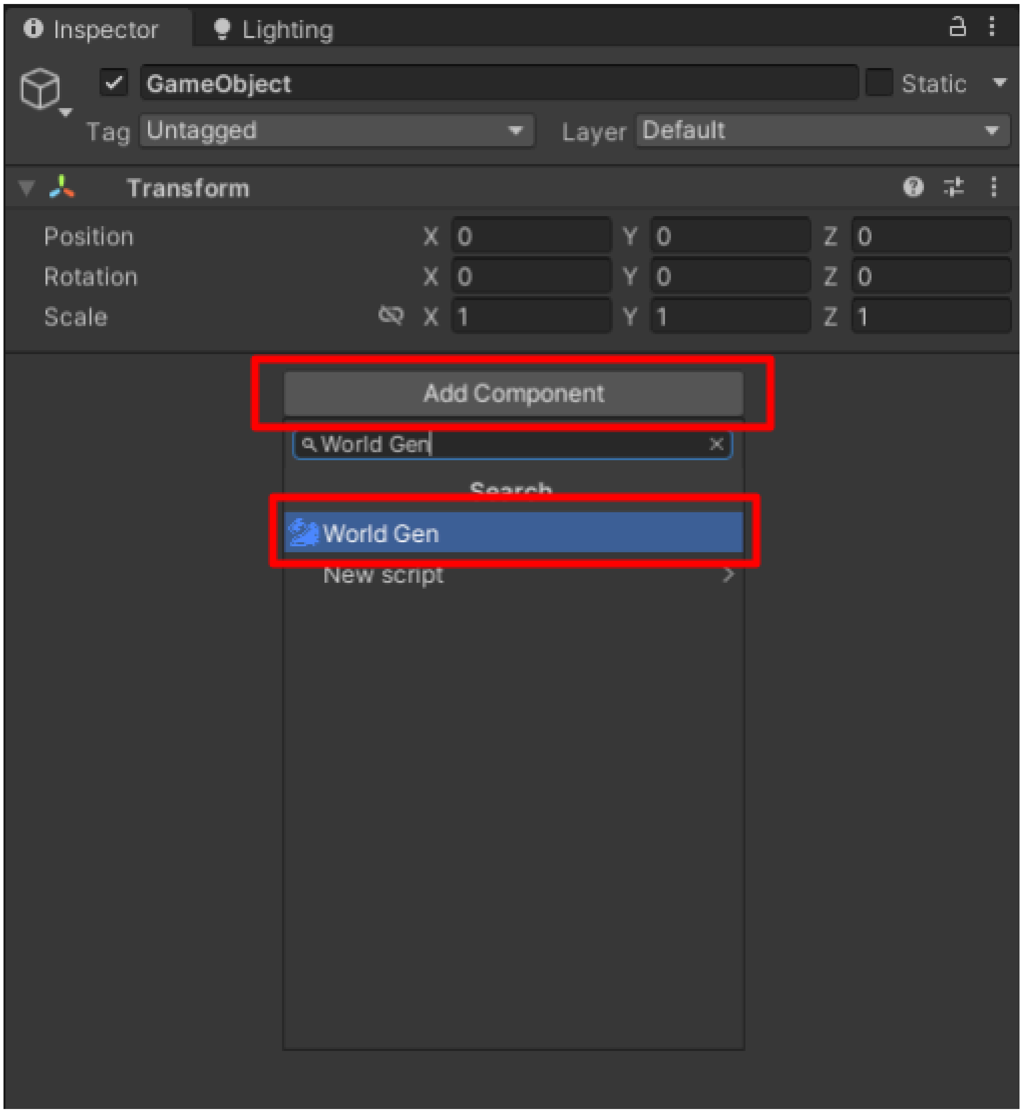
Layer Default

Position X 0 Y 0 Z 0

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

3) Click “Add Component” and search for “World Gen”



☑ GameObject ☐ Static

Tag Untagged

Layer Default

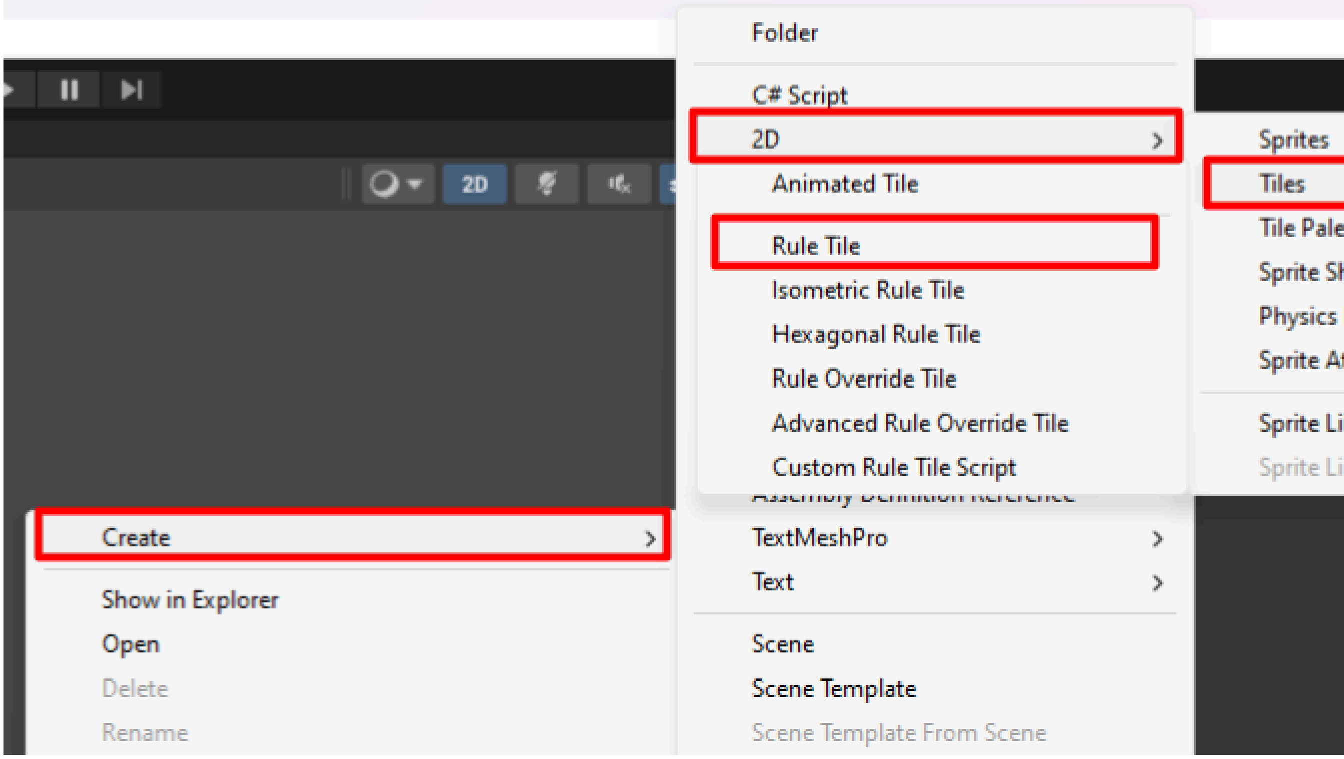
Transform						
Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

World Gen

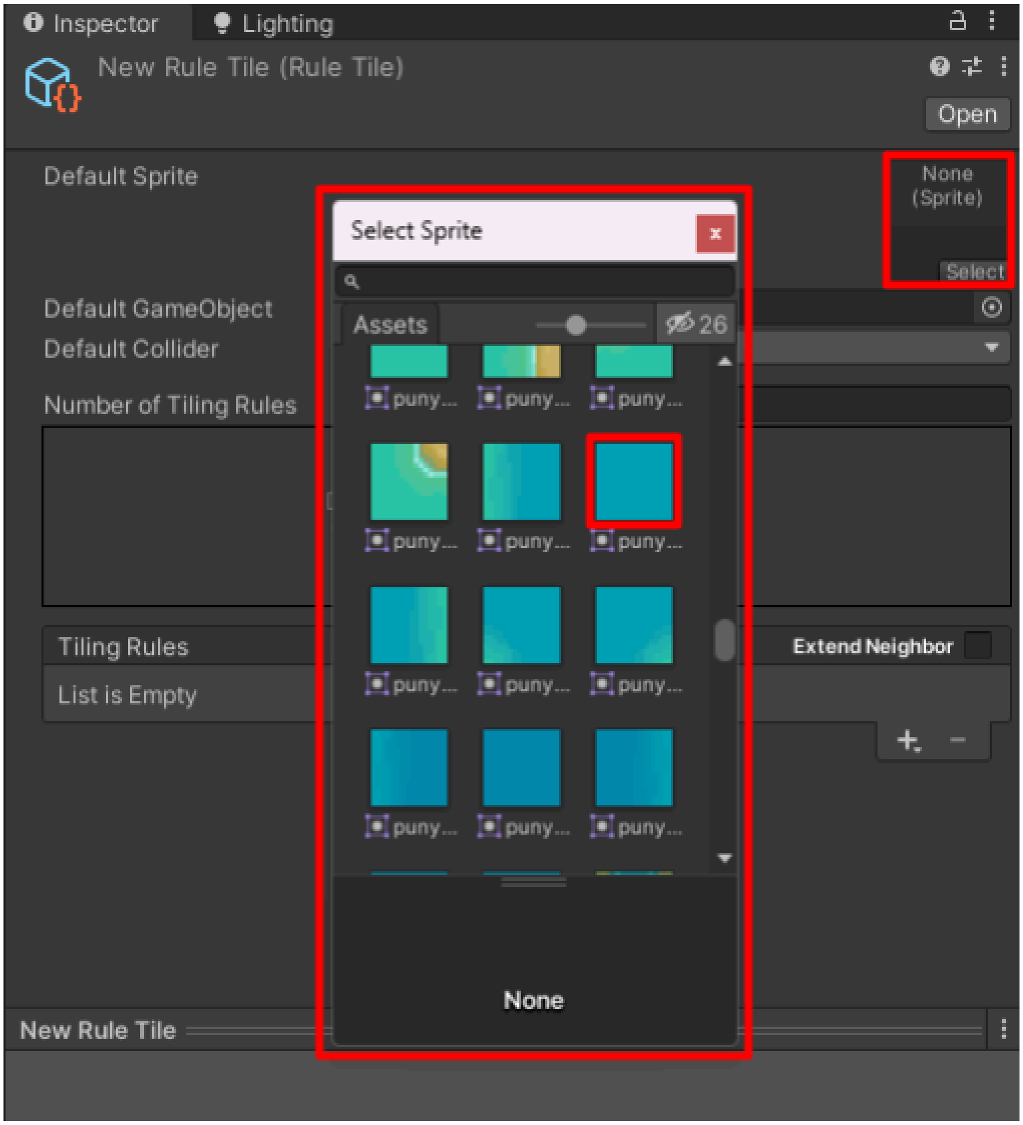
Great! You have everything set up! Next, you just need to create some rule tiles.

RULE TILES

1) To create a rule tile, go to Assets>Create>2D>Tiles>Rule Tile

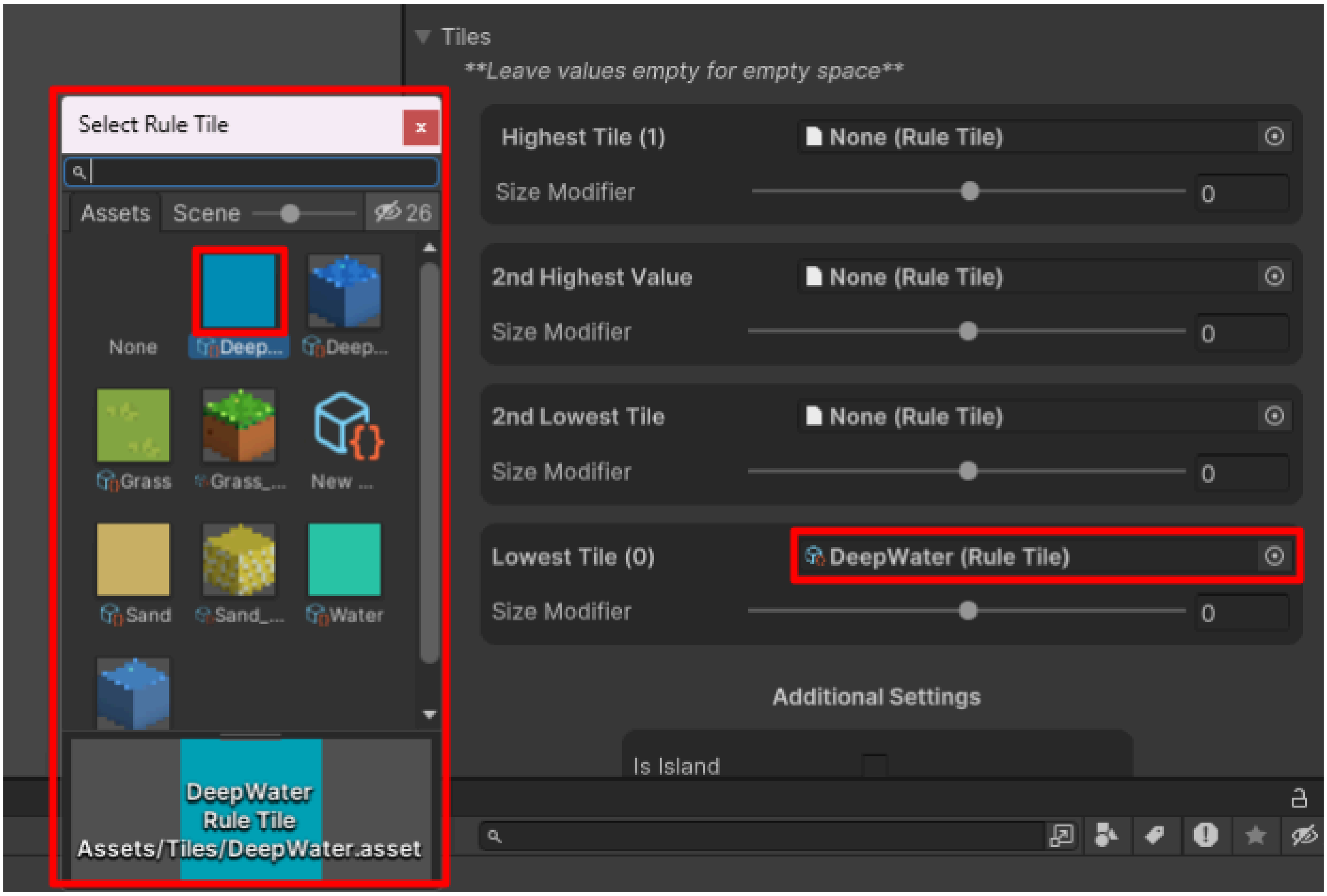


2) In your Rule Tile, change the Default Sprite to a sprite of your choosing



START GENERATING!

Back in the World Generator we set up earlier, add whatever rule tiles you added to the slots available.



Tiles
Leave values empty for empty space

Highest Tile (1) None (Rule Tile)
Size Modifier 0

2nd Highest Value None (Rule Tile)
Size Modifier 0

2nd Lowest Tile None (Rule Tile)
Size Modifier 0

Lowest Tile (0) DeepWater (Rule Tile)
Size Modifier 0

Additional Settings

Is Island ☐

ABOUT THE TILES:

HIGHEST TILE: This is what would be 1 on a noise map. This is what you would want as land.
LOWEST TILE: This is what would be 0 on a noise map. This is what you would want as water.

SECOND HIGHEST TILE: This is what you would want as sand.
SECOND LOWEST TILE: This is what you would want as shallow water.

NOTE: You can also leave these values blank if you want transparency! (Good for cave generation-- just make sure to add a tilemap collider!)

SETTINGS:

- Noise Type: You can switch between Simplex, Perlin, or a combination of the two
- Width and Height: Pretty self-explanatory
- Scale Value: Zooms in or out
- Size Modifier (under tiles): Changes how much of each tile is shown
- Is Island: Do you want to generate an island?
- Island Fallout Size: How big do you want your island?
- Random Seed: Do you want to generate at random? Or do you want to type in your seed?
- Seed: Whatever number you type in will be the seed if Random Seed is selected.

Now all you need to do is click generate!

USING THE EASYWORLDGEN2D NAMESPACE:

You can use certain functions in your own custom code!
All you need to do is include the EasyWorldGen2D namespace when creating a script.

To do this just type

```
using EasyWorldGen2D;
```

at the top of your script are all the other namespaces.

FUNCTIONS

Here are some of the functions you can use and to use them:

```
WorldGenFuntion.SetUp(Transform objectTransform);

//If there is already a child Grid with a Tilemap for a child on the
transform, this function creates a new Grid with a child Tilemap on
which to place our tiles. This Grid is parented to a transform
(object transform).
```

```
WorldGenFunction.Generate(Tiles tileToUse, TileAndGenerationSettings
settings, Tilemap tilemap, int noiseType, int seed, float
scaleFactor, int noiseModifier, bool isIsland);
```

```
//This does exactly what you expect it to do-- Generate our world!
```

In order to be able to fully use WorldGenFunction.Generate(); you will need to understand these classes:

```
public class Tiles
{
      public RuleTile highestValue;
      public RuleTile secondHighestValue;
      public RuleTile secondLowestValue;
      public RuleTile lowestValue;
}
```

```
public class TileAndGenerationSettings
{
      public int width;
      public int height;
      public float islandFalloutSize;

      public bool isUsingRandomSeed;
      public int seed;

      public float highestValue_size;
      public float lowestValue_size;
      public float secondHighestValue_size;
      public float secondLowestValue_size;
}
```

In order to change any of the variables inside these classes, you just need to create a new instance of each class and assign whatever variable you want:

```
//Here is an example of how to change the values in Tiles (this
method will work with ALL classes)

Tiles tiles = new Tiles();
tiles.lowestValue = lowestValue;
tiles.highestValue = highestValue;
tiles.secondHighestValue = secondHighestValue;
tiles.secondLowestValue = secondLowestValue;
```

```
//Then you would put tiles as the argument for each Tiles parameter
(for example, you could put tiles in the tilesToUse parameter in the
Generate(); function).

The same method would apply to any other class.
```

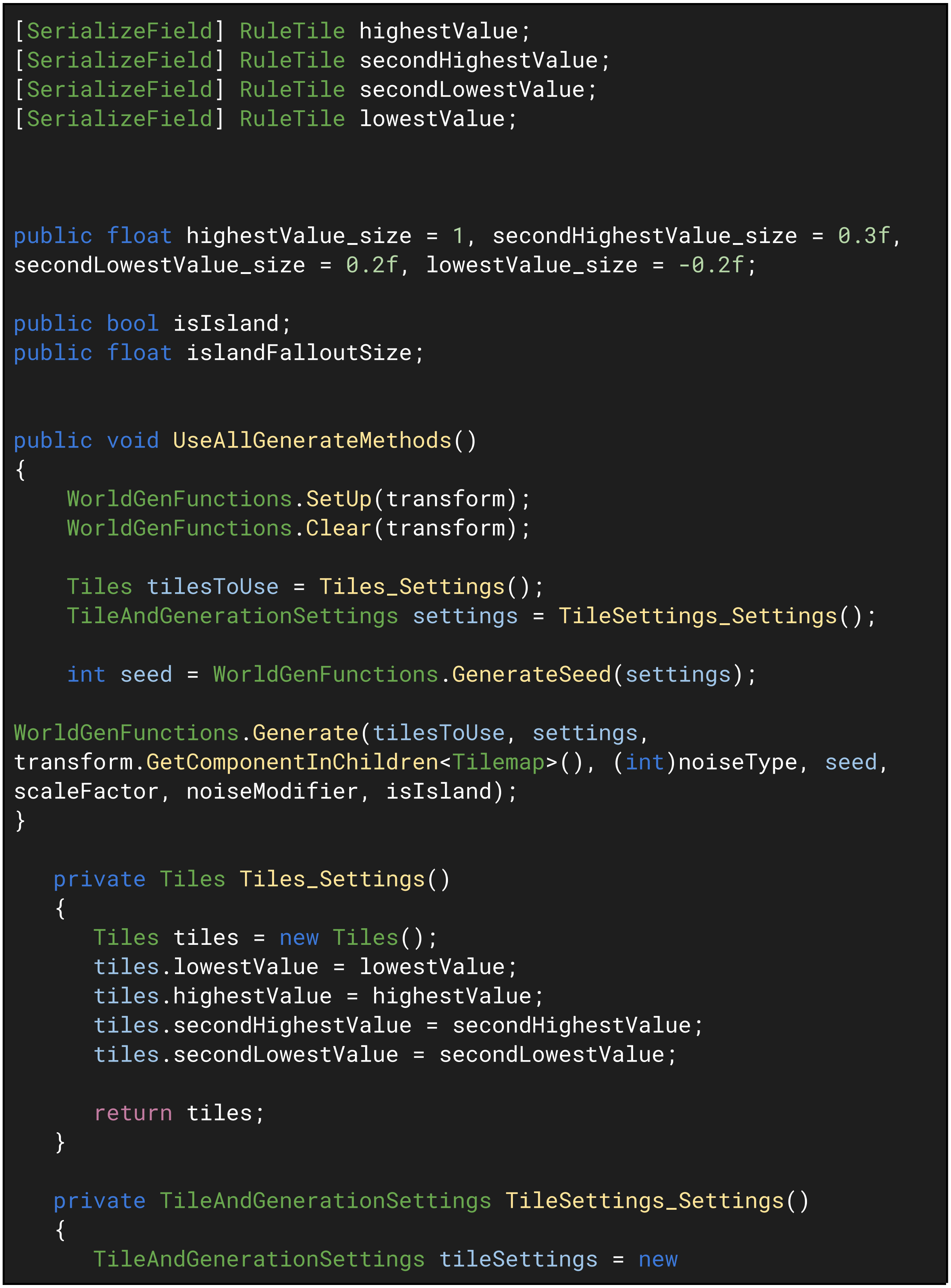
Here are the last two functions:

```
WorldGenFunction.GenerateSeed(TileAndGenerationSettings settings);

//This generates a unique seed if settings.isUsingRandom seed is
true, else you’ll have to set it yourself.

//You would use the above method to create a new instance of
TileAndGenerationSettings, modify the variables, and pass this
instance into the parameter.
```

```
WorldGenFunction.Clear(Transform objTransform);

//This clears the Tilemap. Make sure to put this AFTER the SetUp();
function.

//Just put the transform of the object that has the Grid with the
Tilemap as a child.
```

Here is how all of these functions can work together:

```
public int width = 100, height = 100;
public float scaleFactor = 25;
public int noiseModifier = 1;

public enum NoiseType { Simplex, Perlin, Both}
public NoiseType noiseType;

public int _seed;
public bool randomSeed;
```



```
[SerializeField] RuleTile highestValue;
[SerializeField] RuleTile secondHighestValue;
[SerializeField] RuleTile secondLowestValue;
[SerializeField] RuleTile lowestValue;



public float highestValue_size = 1, secondHighestValue_size = 0.3f,
secondLowestValue_size = 0.2f, lowestValue_size = -0.2f;

public bool isIsland;
public float islandFalloutSize;


public void UseAllGenerateMethods()
{
    WorldGenFunctions.SetUp(transform);
    WorldGenFunctions.Clear(transform);

    Tiles tilesToUse = Tiles_Settings();
    TileAndGenerationSettings settings = TileSettings_Settings();

    int seed = WorldGenFunctions.GenerateSeed(settings);

WorldGenFunctions.Generate(tilesToUse, settings,
transform.GetComponentInChildren<Tilemap>(), (int)noiseType, seed,
scaleFactor, noiseModifier, isIsland);
}

   private Tiles Tiles_Settings()
   {
      Tiles tiles = new Tiles();
      tiles.lowestValue = lowestValue;
      tiles.highestValue = highestValue;
      tiles.secondHighestValue = secondHighestValue;
      tiles.secondLowestValue = secondLowestValue;

      return tiles;
   }

   private TileAndGenerationSettings TileSettings_Settings()
   {
      TileAndGenerationSettings tileSettings = new
```

```
        TileAndGenerationSettings();

        tileSettings.width = width;
        tileSettings.height = height;
        tileSettings.islandFalloutSize = islandFalloutSize;

        tileSettings.isUsingRandomSeed = randomSeed;
        tileSettings.seed = _seed;

        tileSettings.highestValue_size = highestValue_size;
        tileSettings.lowestValue_size = lowestValue_size;
        tileSettings.secondHighestValue_size = secondHighestValue_size;
        tileSettings.secondLowestValue_size = secondLowestValue_size;

        return tileSettings;
    }
```

You can look inside the included scripts to further see any functions that you could use!

NOTES

You can create multiple GameObjects with the WorldGen component on them and you can change their positions.