2D Neon Grids (Shader Graph) for Unity

2D Neon Grids is a **Shader Graph** asset for **Unity**. It allows you to generate 2D grid patterns with two texture layers, customisable color, as well as many different effects like distortion, scan-line, rotation, movement, and tiling.

You you can apply the grid patterns to your Unity games/projects by using **Sprites** or **Quads**. Grids can also be masked off using 2D Sprite Masks.

Use bloom to really make them pop and get those retro neon effects going.

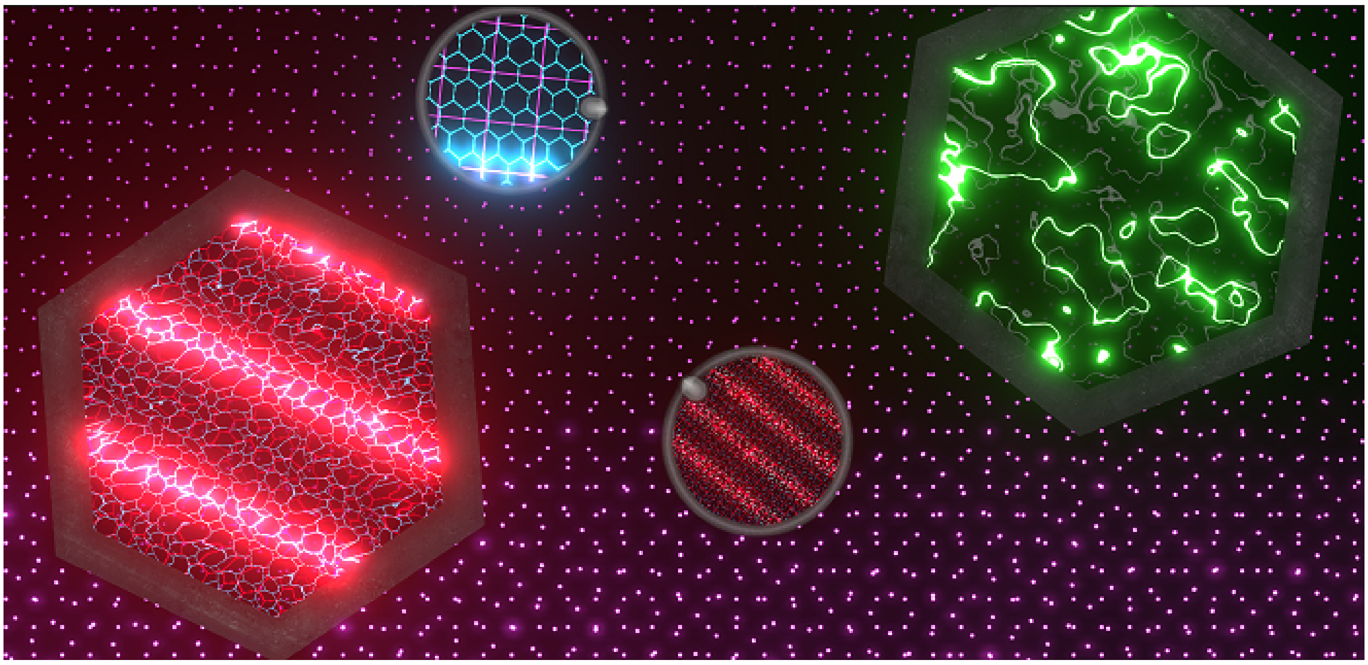
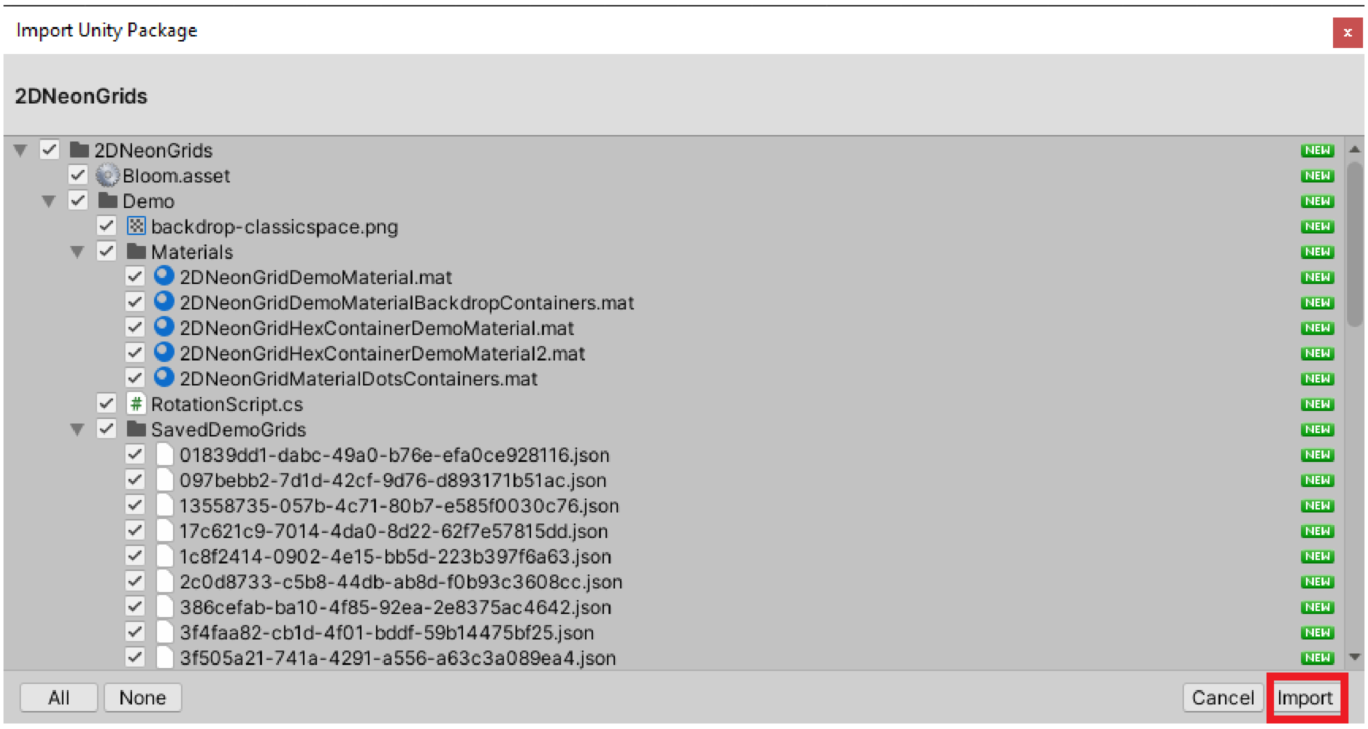


Table of Contents

Getting Started

To get started, import the 2D Neon Grids asset into your project.

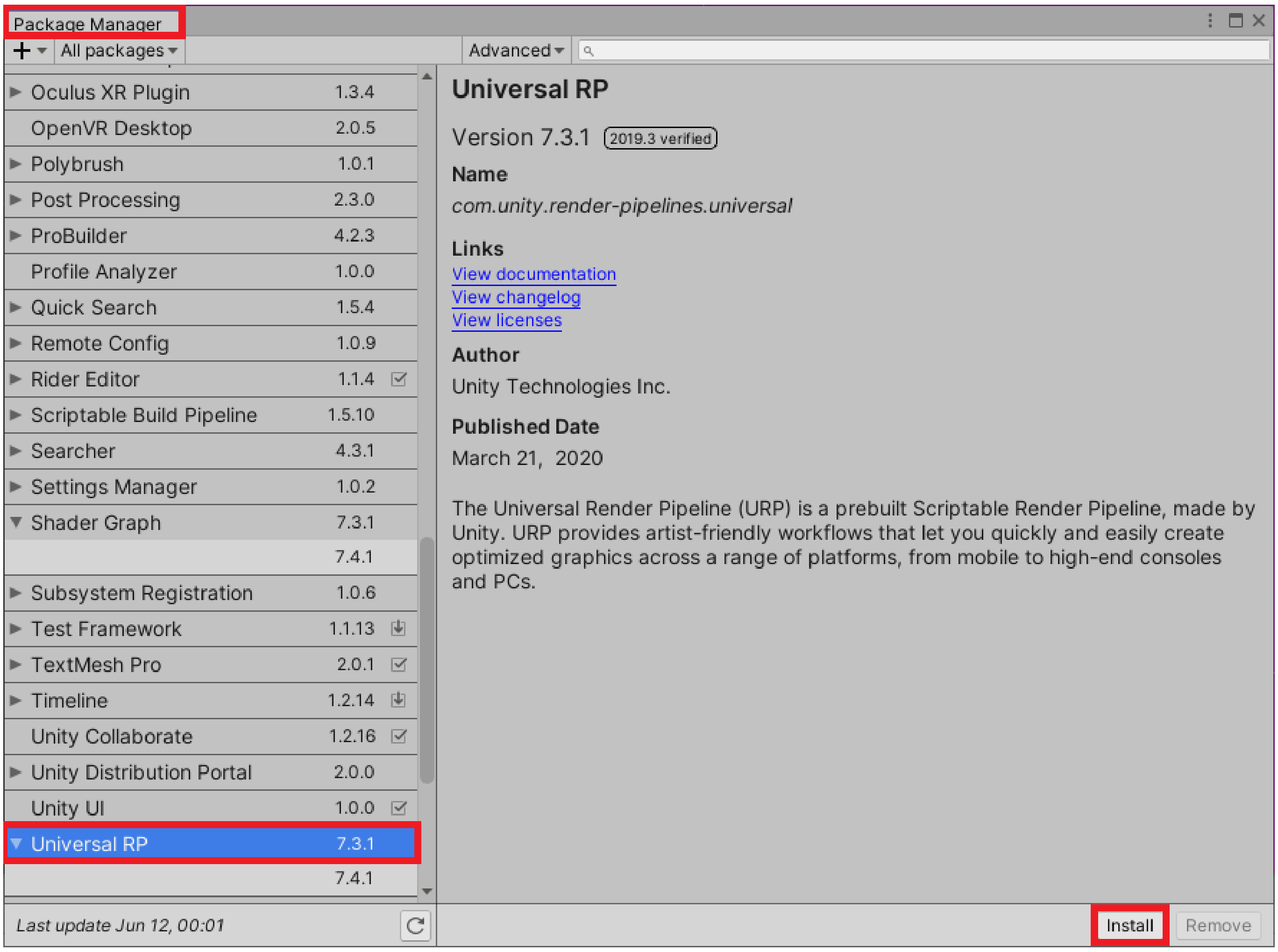

Import Unity Package

2DNeonGrids

```
2DNeonGrids
  Bloom.asset
  Demo
    backdrop-classicspace.png
    Materials
      2DNeonGridDemoMaterial.mat
      2DNeonGridDemoMaterialBackdropContainers.mat
      2DNeonGridHexContainerDemoMaterial.mat
      2DNeonGridHexContainerDemoMaterial2.mat
      2DNeonGridMaterialDotsContainers.mat
    RotationScript.cs
    SavedDemoGrids
      01839dd1-dabc-49a0-b76e-efa0ce928116.json
      097bebb2-7d1d-42cf-9d76-d893171b51ac.json
      13558735-057b-4c71-80b7-e585f0030c76.json
      17c621c9-7014-4da0-8d22-62f7e57815dd.json
      1c8f2414-0902-4e15-bb5d-223b397f6a63.json
      2c0d8733-c5b8-44db-ab8d-f0b93c3608cc.json
      386cefab-ba10-4f85-92ea-2e8375ac4642.json
      3f4faa82-cb1d-4f01-bddf-59b14475bf25.json
      3f505a21-741a-4291-a556-a63c3a089ea4.json
```

If you don't already have the following packages installed, open the Unity Package Manager and install these:

- Universal RP (Universal Render Pipeline)
- Post Processing



Oculus XR Plugin	1.3.4
OpenVR Desktop	2.0.5
Polybrush	1.0.1
Post Processing	2.3.0
ProBuilder	4.2.3
Profile Analyzer	1.0.0
Quick Search	1.5.4
Remote Config	1.0.9
Rider Editor	1.1.4
Scriptable Build Pipeline	1.5.10
Searcher	4.3.1
Settings Manager	1.0.2
Shader Graph	7.3.1
	7.4.1
Subsystem Registration	1.0.6
Test Framework	1.1.13
TextMesh Pro	2.0.1
Timeline	1.2.14
Unity Collaborate	1.2.16
Unity Distribution Portal	2.0.0
Unity UI	1.0.0
Universal RP	7.3.1
	7.4.1

Last update Jun 12, 00:01

Universal RP

Version 7.3.1 (2019.3 verified)

Name
com.unity.render-pipelines.universal

Links
View documentation
View changelog
View licenses

Author
Unity Technologies Inc.

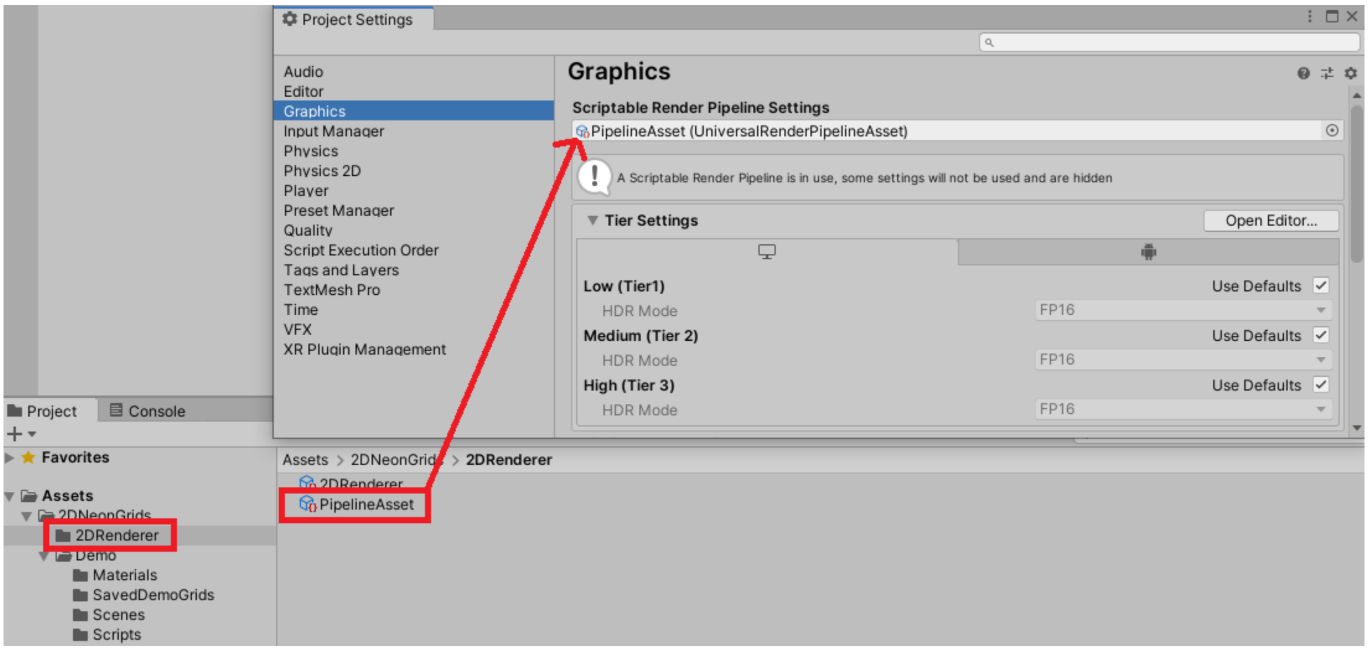
Published Date
March 21, 2020

The Universal Render Pipeline (URP) is a prebuilt Scriptable Render Pipeline, made by Unity. URP provides artist-friendly workflows that let you quickly and easily create optimized graphics across a range of platforms, from mobile to high-end consoles and PCs.

Next you'll need to configure your scriptable render pipeline asset (if you don't already have one set).

Open **Edit** > **Project Settings** > **Graphics**

Drag and drop the **PipelineAsset** from **Assets/2DNeonGrids/2DRenderer** into the **Scriptable Render Pipeline Settings** slot in the Graphics settings for your project.



Graphics

Scriptable Render Pipeline Settings
PipelineAsset (UniversalRenderPipelineAsset)

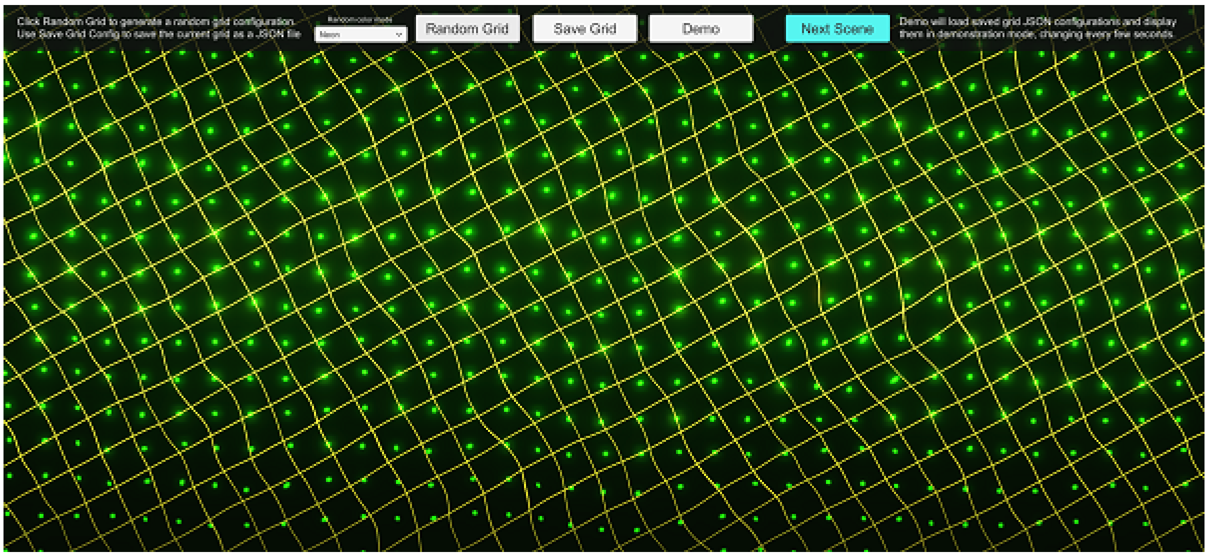
A Scriptable Render Pipeline is in use, some settings will not be used and are hidden

Tier Settings

Low (Tier1) Use Defaults ✓
HDR Mode FP16
Medium (Tier 2) Use Defaults ✓
HDR Mode FP16
High (Tier 3) Use Defaults ✓
HDR Mode FP16

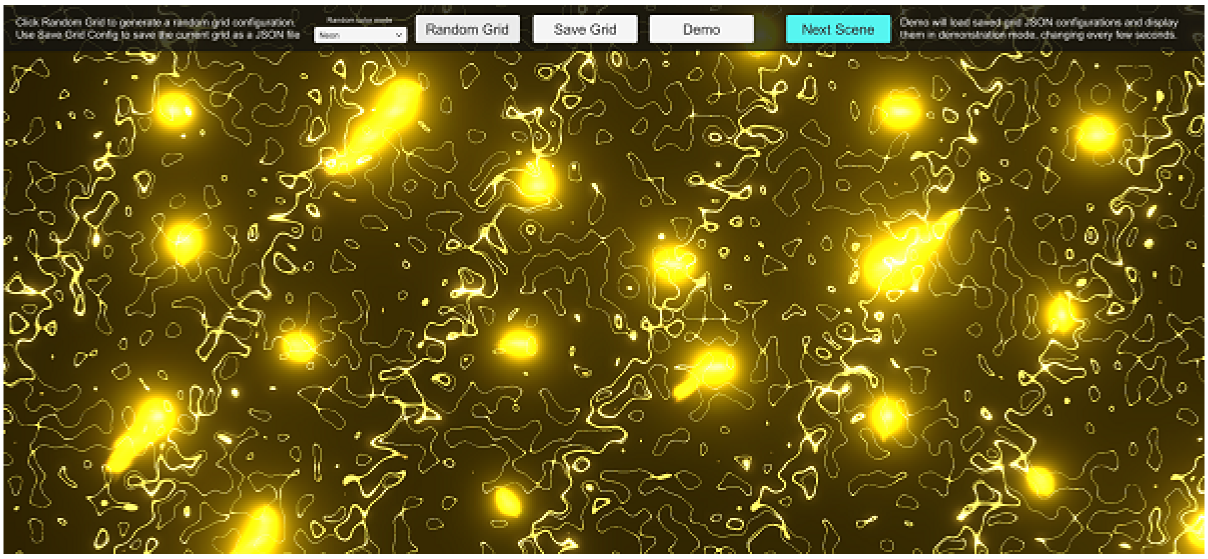
At this point you should now be ready to open and run the included Demo scenes.

Open **Assets/2DNeonGrids/Demo/Scenes/DemoSpriteMasks.unity** or **Assets/2DNeonGrids/Demo/Scenes/DemoRandomizer.unity** to check out the included demo scenes.



Click Random Grid to generate a random grid configuration.
Use Save Grid Config to save the current grid as a JSON file

Demo will load saved grid JSON configurations and display them in demonstration mode, changing every few seconds.



Click Random Grid to generate a random grid configuration.
Use Save Grid Config to save the current grid as a JSON file

Demo will load saved grid JSON configurations and display them in demonstration mode, changing every few seconds.

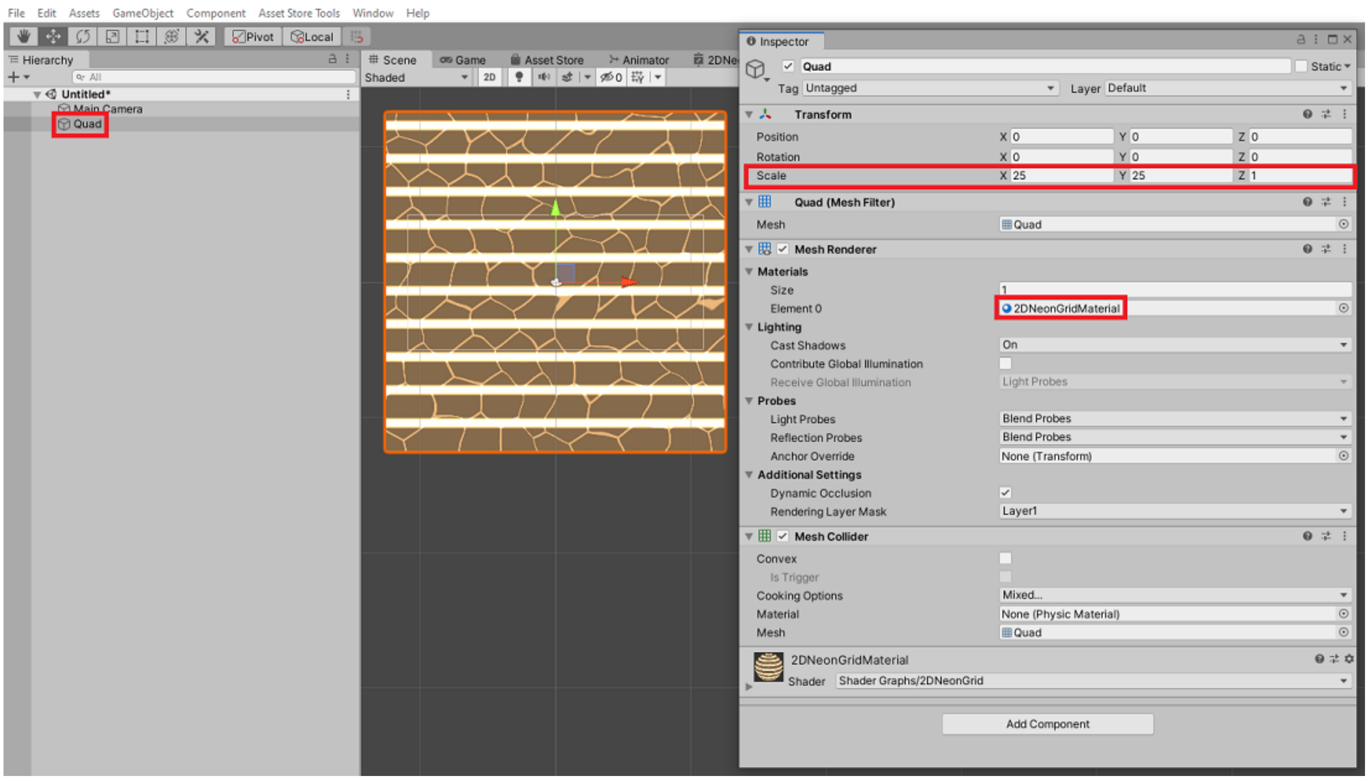
Grid Material Configuration

To configure your grid you'll need:

- A 2D **Sprite** or **Quad** GameObject
- A **Material** set to use the **2DNeonGrids** shader graph shader (there is one included with the asset already)

Example

- In your scene, create a new Quad GameObject.
- With **Quad** selected, use the Inspector to assign the **Assets/2DNeonGrids/Shaders/2DNeonGridMaterial.material** asset to the **Materials Element 0** slot.



```
Inspector
Quad                                   Static
Tag Untagged          Layer Default
Transform
  Position   X 0    Y 0    Z 0
  Rotation   X 0    Y 0    Z 0
  Scale      X 25   Y 25   Z 1
Quad (Mesh Filter)
  Mesh       Quad
Mesh Renderer
  Materials
    Size                              1
    Element 0                         2DNeonGridMaterial
  Lighting
    Cast Shadows                      On
    Contribute Global Illumination
    Receive Global Illumination       Light Probes
  Probes
    Light Probes                      Blend Probes
    Reflection Probes                 Blend Probes
    Anchor Override                   None (Transform)
  Additional Settings
    Dynamic Occlusion                 ✓
    Rendering Layer Mask              Layer1
Mesh Collider
  Convex
    Is Trigger
  Cooking Options                     Mixed...
  Material                            None (Physic Material)
  Mesh                                Quad
2DNeonGridMaterial
  Shader  Shader Graphs/2DNeonGrid
Add Component
```

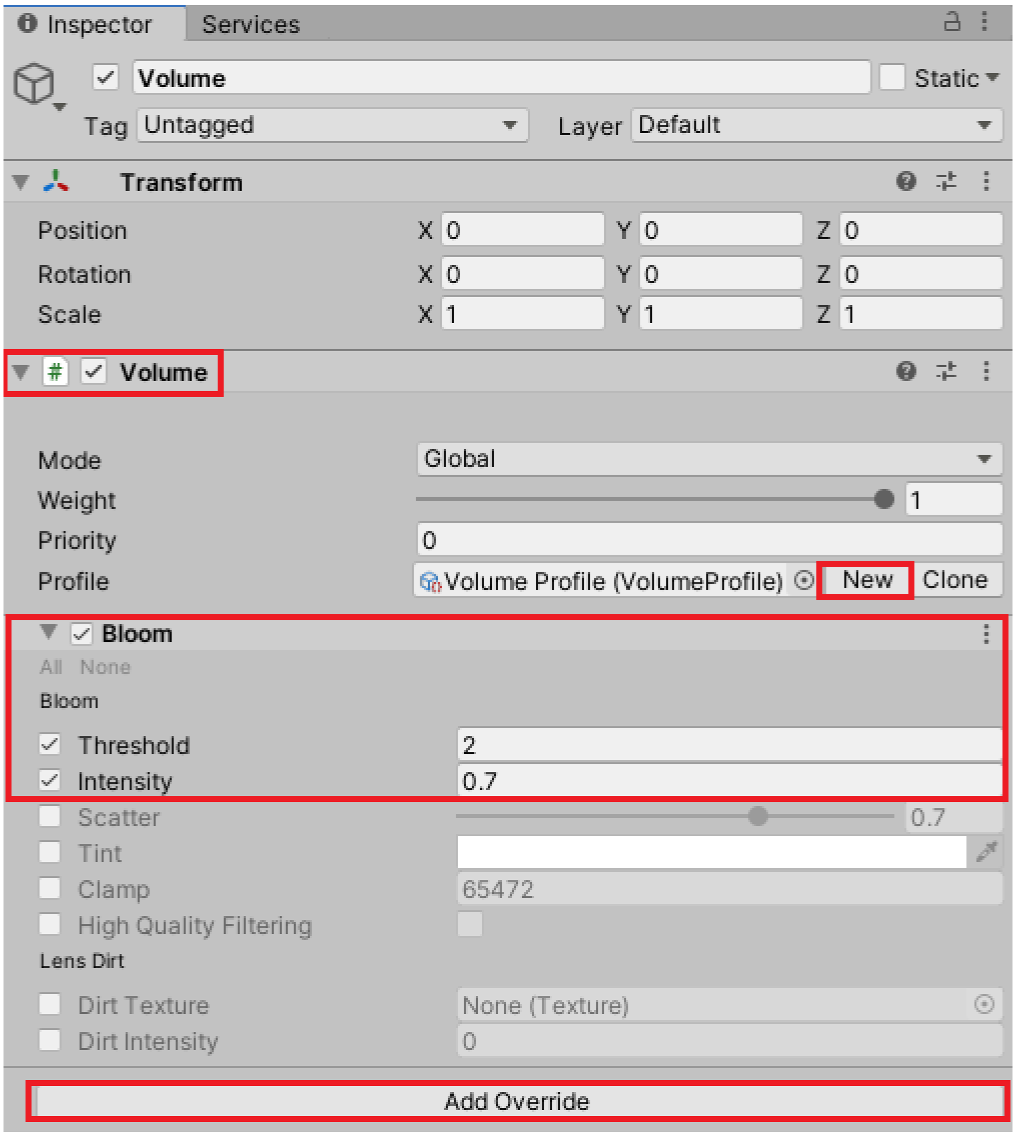
Adding Neon Bloom / Glow

Create a new **GameObject** in your Scene **Hierarchy** and name it **Volume**. Use the Inspector to add a **Volume** component. In the **Volume** component Inspector options, click **New** next to the **Profile** field. A new profile called **Volume Profile (VolumeProfile)** is created.

Click the **Add Override** button and type **Bloom** to add a Bloom volume effect.

Configure the **Bloom** settings to have:

- **Threshold: 2**
- **Intensity: 0.7**



Volume
Tag Untagged Layer Default

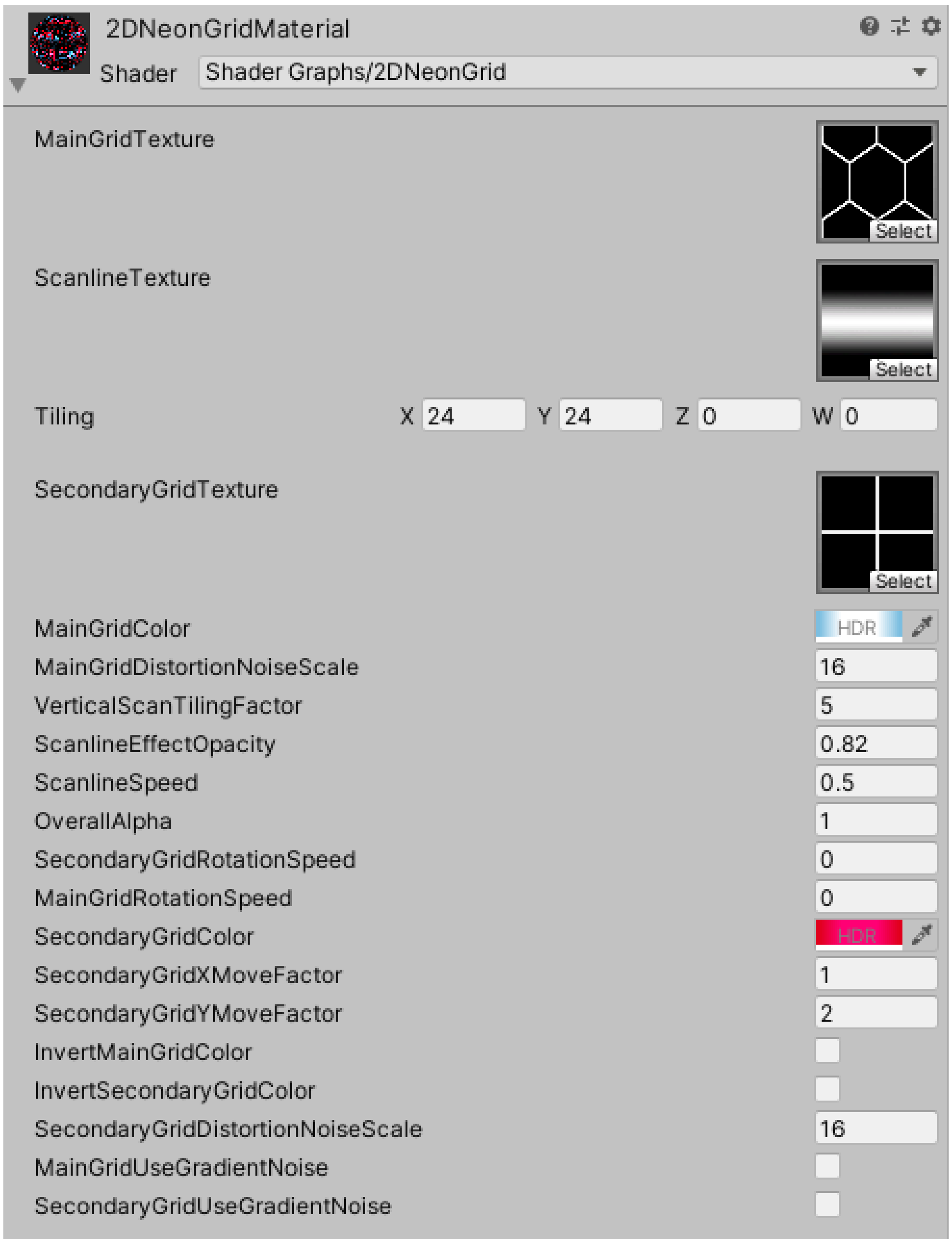
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Volume
Mode Global
Weight 1
Priority 0
Profile Volume Profile (VolumeProfile)

Bloom
Bloom
Threshold 2
Intensity 0.7
Scatter 0.7
Tint
Clamp 65472
High Quality Filtering
Lens Dirt
Dirt Texture None (Texture)
Dirt Intensity 0

Configure grid shader settings

Rollout the **2DNeonGridMaterial Shader** section in the Inspector (near the bottom), and you can begin customizing your grid with the properties exposed here.



2DNeonGridMaterial

Shader Shader Graphs/2DNeonGrid

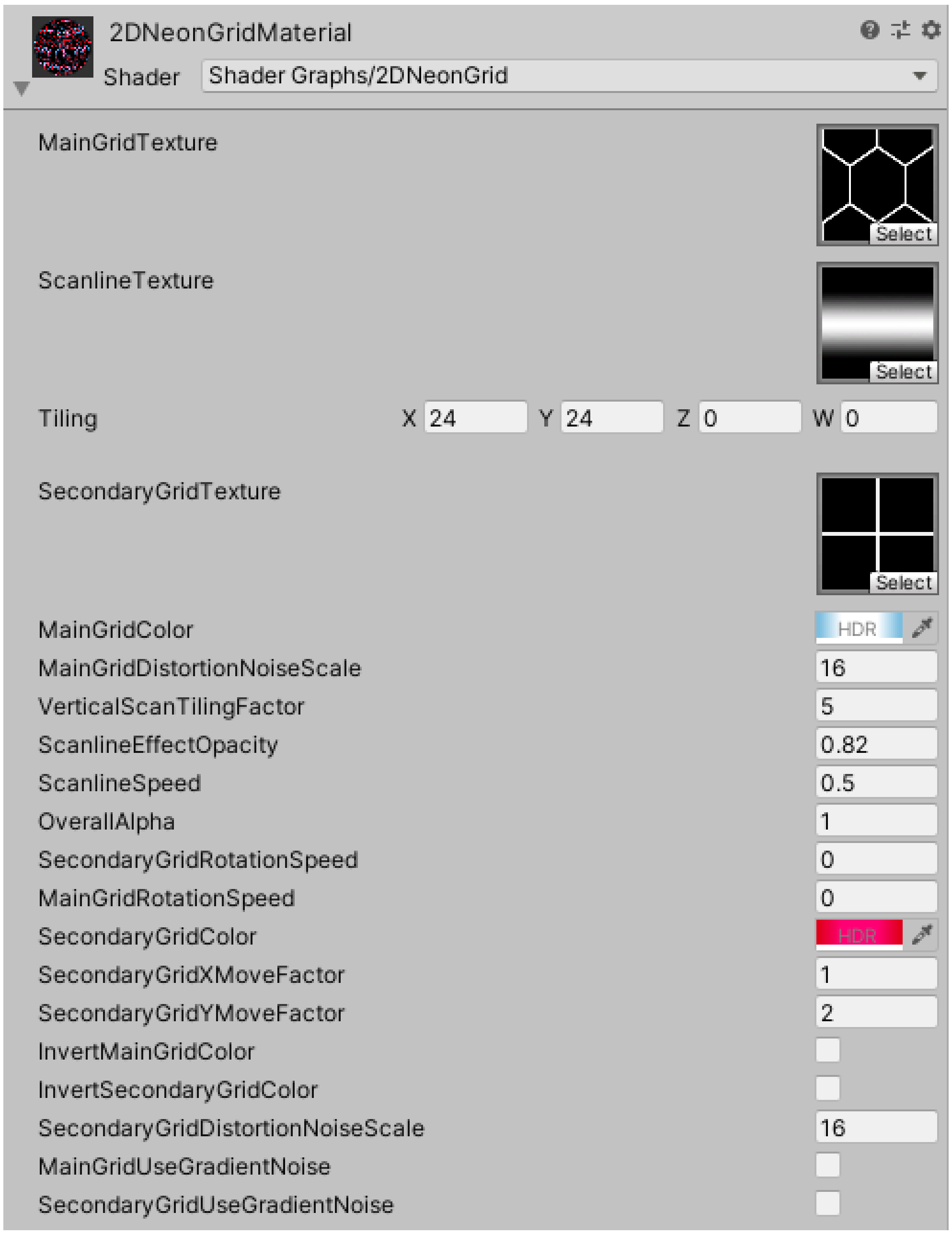
MainGridTexture	Select
ScanlineTexture	Select
Tiling	X 24 Y 24 Z 0 W 0
SecondaryGridTexture	Select
MainGridColor	HDR
MainGridDistortionNoiseScale	16
VerticalScanTilingFactor	5
ScanlineEffectOpacity	0.82
ScanlineSpeed	0.5
OverallAlpha	1
SecondaryGridRotationSpeed	0
MainGridRotationSpeed	0
SecondaryGridColor	HDR
SecondaryGridXMoveFactor	1
SecondaryGridYMoveFactor	2
InvertMainGridColor	☐
InvertSecondaryGridColor	☐
SecondaryGridDistortionNoiseScale	16
MainGridUseGradientNoise	☐
SecondaryGridUseGradientNoise	☐

The following are descriptions of what each exposed setting does:

- **MainGridTexture** - the primary, or main grid texture which is set to **Wrap Mode** '**Repeat**'
- **SecondaryGridTexture** - the primary, or main grid texture which is set to **Wrap Mode** '**Repeat**'
- **ScanlineTexture** - the texture which is used for the optional scanline scrolling effect on the grid.
- **Tiling** - a Vector4 property which is used to control the **X** and **Y** tiling of the grid textures.
- **MainGridColor** - the color used for the Main Grid texture. Use an **HDR** color with Intensity to increase bloom/glow effect.
- **SecondaryGridColor** - the color used for the Main Grid texture. Use an **HDR** color with Intensity to increase bloom/glow effect.
- **MainGridDistortionNoiseScale** - used to increase the noise texture scaling effect (Simple or Gradient Noise) on the main grid.
- **SecondaryGridDistortionNoiseScale** - used to increase the noise texture scaling effect (Simple or Gradient Noise) on the secondary grid.
- **ScanlineEffectOpacity** - used to increase or decrease the scanline 'area of effect'.
- **ScanlineSpeed** - increase or decrease downward/upward scanline movement.

- **VerticalScanTilingFactor** - used to increase or decrease the number of scanlines. You could for example use this to create CRT monitor-like effects.
- **OverallAlpha** - an overall alpha or opacity setting to make the final grid texture see-through or opaque.
- **MainGridRotationSpeed** - used to rotate the main grid texture.
- **SecondaryGridRotationSpeed** - used to rotate the secondary grid texture.
- **SecondaryGridXMoveFactor** - used to apply a sine-like movement to the secondary grid texture on the X-axis.
- **SecondaryGridYMoveFactor** - used to apply a sine-like movement to the secondary grid texture on the Y-axis.
- **InvertMainGridColor** - inverts the main grid texture color.
- **InvertSecondaryGridColor** - inverts the secondary grid texture color.
- **MainGridUseGradientNoise** - toggle this on to use Gradient (Perlin) Noise instead of Simple Noise on the main grid texture.
- **SecondaryGridUseGradientNoise** - toggle this on to use Gradient (Perlin) Noise instead of Simple Noise on the secondary grid texture.

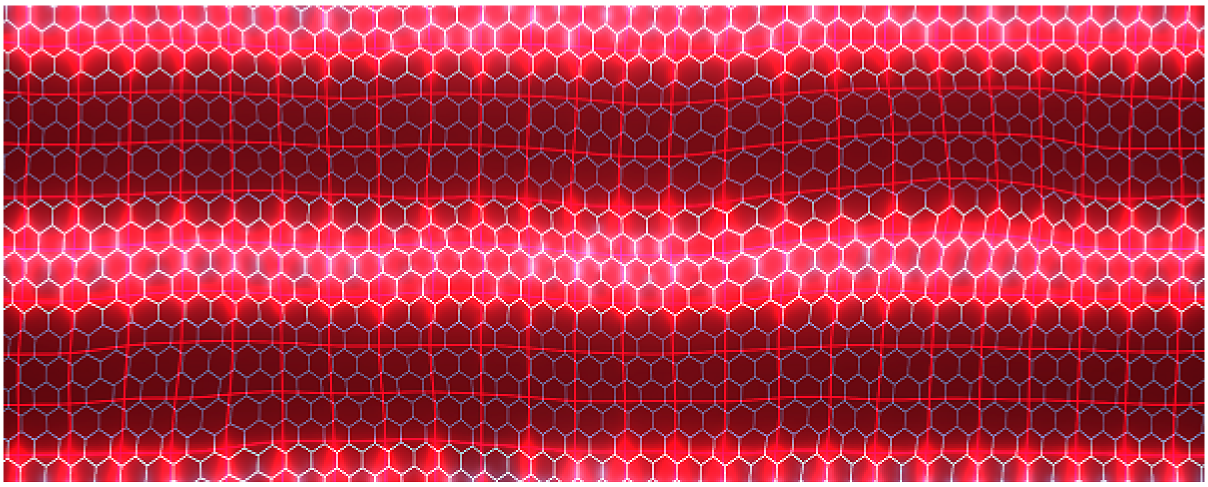
Configure the grid settings as follows for a sample grid:



2DNeonGridMaterial
Shader: Shader Graphs/2DNeonGrid

MainGridTexture	
ScanlineTexture	
Tiling	X 24 Y 24 Z 0 W 0
SecondaryGridTexture	
MainGridColor	HDR
MainGridDistortionNoiseScale	16
VerticalScanTilingFactor	5
ScanlineEffectOpacity	0.82
ScanlineSpeed	0.5
OverallAlpha	1
SecondaryGridRotationSpeed	0
MainGridRotationSpeed	0
SecondaryGridColor	HDR
SecondaryGridXMoveFactor	1
SecondaryGridYMoveFactor	2
InvertMainGridColor	☐
InvertSecondaryGridColor	☐
SecondaryGridDistortionNoiseScale	16
MainGridUseGradientNoise	☐
SecondaryGridUseGradientNoise	☐

Run the scene. Your Quad grid texture should look like the following, and be animating using the grid's various effects.



You can play around with different effects by trying out different textures for the Main and Secondary Grid, as well as Scanline texture settings. These, in combination with the exposed properties listed above can result in thousands of different grid configurations. Use the **DemoRandomizer** example scene and click the **Random** button multiple times, with different color schemes selected to get some inspiration.

Saving and Loading Grid Configurations at Runtime

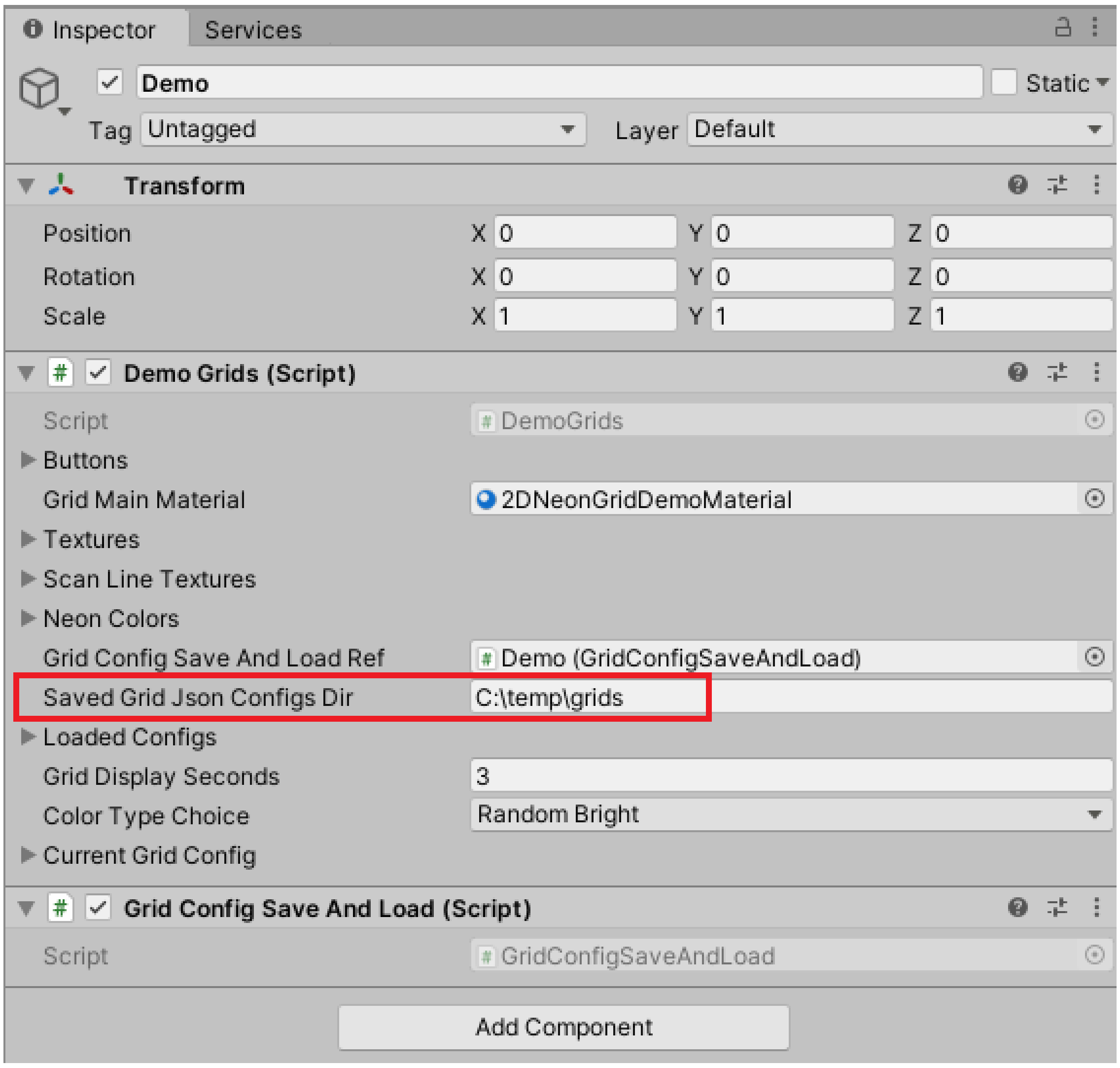
Grids configurations from the **DemoRandomizer.unity** scene can be saved to JSON format on disk and dynamically re-loaded at runtime. To save a Grid that you like (or have customised in the Editor at runtime, do the following:

- Click **Save Grid**
- Open your project's **Assets** folder and you'll see a randomly (GUID) named JSON file. This file contains the grid's configuration serialized to file.

To load grids from previously saved JSON format at runtime, move the JSON file into **Assets/2DNeonGrids/Demo/SavedDemoGrids** and run the demo scene again. Click the Demo button and all Grid Configuration JSON files in that folder will be loaded. The scene will then cycle through each one every 3 seconds.

If you would like to learn how to dynamically configure your 2D Neon Grids shader materials at runtime / from code, then take a look at the **DemoGrids.cs** script, and specifically the `LoadGridConfig` method.

Note: The asset includes a bunch of demo saved grid configurations in the **2DNeonGrids/Demo/SavedDemoGrids** asset folder. By default the **DemoRandomizer** scene will load the grids from this path in your project, however you can pinpoint a different location to load the JSON grid configuration files by entering a path value in the **Saved Grid Json Configs Dir** field.



Inspector | Services

Demo | Static
Tag: Untagged | Layer: Default

Transform

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

Demo Grids (Script)

Script	DemoGrids
▶ Buttons	
Grid Main Material	2DNeonGridDemoMaterial
▶ Textures	
▶ Scan Line Textures	
▶ Neon Colors	
Grid Config Save And Load Ref	Demo (GridConfigSaveAndLoad)
Saved Grid Json Configs Dir	C:\temp\grids
▶ Loaded Configs	
Grid Display Seconds	3
Color Type Choice	Random Bright
▶ Current Grid Config	

Grid Config Save And Load (Script)

Script	GridConfigSaveAndLoad

Support

If you need help or support, please get in touch using the contact form here: Contact or e-mail: apps@shogan.co.uk