

Traffy

Weather & Background Controller

ReadMe

Table of Contents

Sky & Time Controller	2
Time Settings	2
Background Settings.....	2
Stars Settings.....	2
Background Settings	4
Stars Settings	6
Weather Generator	7
Wind Settings.....	7
Weather Elements	7
Wind Settings	8
Weather Elements.....	9
Public Functions	13
Background Generator.....	14
Repeat Build - Troubleshoot/Design Only.....	14
Generation Settings.....	14
Looping	14
Build Over Time.....	14
Outline Settings	14
Camera Position Settings	14
Generation Settings	16
Looping	18
Build Over Time.....	19
Outline Settings	20
Camera Position Settings	21
Saving A Generated Mesh.....	22
Parallax Object.....	23
Settings.....	23
Contact/Troubleshooting.....	24

Sky & Time Controller

This object controls the time and sky settings, as well as the stars. It is required for the Weather Generator.

Time Settings

- Control the overall 24 hour time of the weather
- Set the rate that time increases - including backwards
- Pause the time settings to hold a particular gradient

Background Settings

- Create your own background sky in a gradient, built to be just bigger than your camera (with a multiplier option for use as required)
- Easily control the order-in-layer of your sky background
- Multiple resolution settings for level of detail
- Determine different background shades for different times of the day. The background will automatically update over time to lerp between your different gradients at the appropriate times.

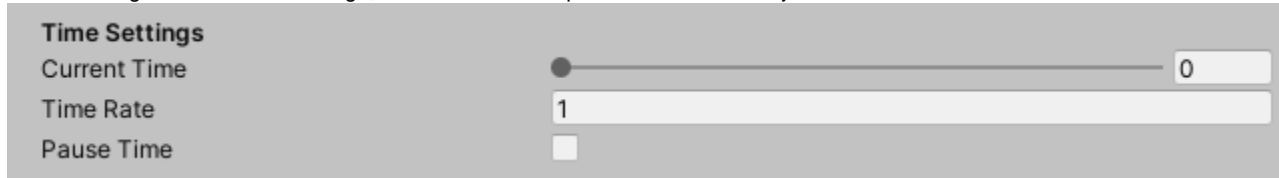
Stars Settings

- Set your own stars object, or use the default provided prefab
- Control the order in layer of the stars as required
- Determine the start and finish of hours appearing according to the time, including an amount of time to slowly fade stars into view
- Create a parallax scale so stars slowly drift with camera movement

For more details on how to use each of these objects, please see the relevant next section.

Time Settings

This section governs the time settings, and how to use this part of the Weather object.



Here you can see there are three specific public variables in the TimeSettings inspector.

Current Time: This is the current time, represented in hours - going from 0 - 23.99. Here you can set the time the scene will start. CurrentTime cannot be changed while in game mode unless time is paused. Pause Time (described below) when true will allow Current Time to be manually changed.

Time Rate: The rate at which time will advance in seconds. This is compared to realtime seconds. For example, a value of 1 will be 1 second and game time equals 1 second of realtime - so the game will advance at the same rate as real life. A value of 3600 (1 hour in seconds), will advance time 1 hour for every second that passes in real time.

Pause Time: When true, time will be paused and cease advancing. Further, to manually update the current time in game mode, you can pause time and change the Current Time before unpausing.

Each of these settings can also be updated by calling this function [SetTime](#).

Background Settings

Background settings are related specifically to the sky background and its changes overtime - and is governed by the following settings on the Weather object:

Background Settings

Spawn Background ☒

Cam Size Multiplier

Background Order In Layer

My Res

▼ **Background Shades**

Size

▼ **Element 0**

Hour

Minute

My Gradient

▼ **Element 1**

Hour

Minute

My Gradient

▼ **Element 2**

Hour

Minute

My Gradient

▼ **Element 3**

Hour

Minute

My Gradient

▼ **Element 4**

Hour

Minute

My Gradient

Spawn Background: When true, the object will spawn the sky when the game starts. This is optional, so if you do not intend to use this feature you can disable it.

Cam Size Multiplier: When building the background, the object looks at the current size of the camera. It will then build the dimensions of the sky proportional to the camera. A value of 1 matches the camera exactly. Values higher or lower will scale appropriately. If you intend to have a camera zooming in and out, it is recommended to use a scale value that will cover all requirements.

Background Order In Layer: Set the order in layer value of the created background. Objects with a higher order in layer appear at the front of the orthographic camera, and lower numbers appear at the back. As such, it is recommended to give this object a low number.

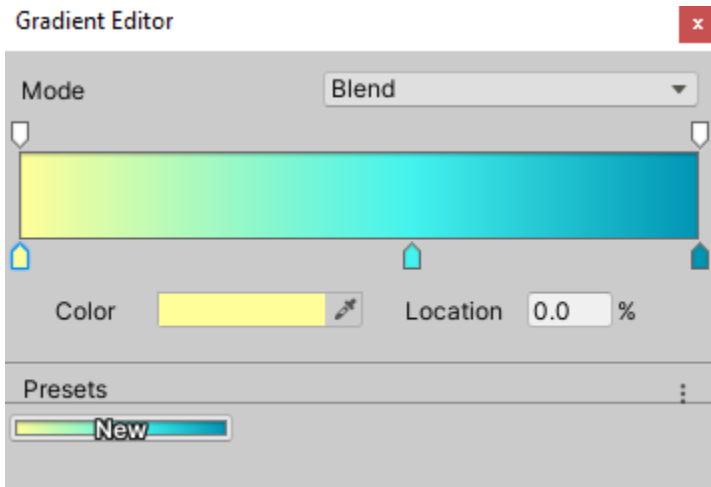
My Res: Set the resolution of the object from a dropdown menu. Higher resolutions increase the visual quality but create a higher processing requirement.

Background Shades: Here you can set the colour of the background sky, and at what time you wish for this shade to appear. In element zero, you can see Hour and Minute are both set to 0. This is equivalent to midnight. At midnight, the background will be the shade shown in My Gradient, with the left side being the low end, and green being the top.

As time progresses, and it reaches Element 1 - it will over time lerp colour to the yellow/blue gradient shown in Element 1.

i Note: Elements do not have to be in time order. The code will loop through each and determine which is next, should your gradients be out of order.

Set the Hour and Minute you want your gradient to appear, and double click the gradient colour to set your own shades:



For any questions regarding how to use the Gradient Editor - please see the Unity Docs.

Shades can be added and removed as required by increasing the Background Shades Size value. Please note, reducing the size will remove the lowest shades. Increasing the value will duplicate the last shade for as many as the size has increased.

Stars Settings

These settings govern spawning a background object such as stars that fade in and out according to the time. It is located on the weather object and is related to the following fields:

Setting	Value
Spawn Stars	<input checked="" type="checkbox"/>
Stars Prefab	StarsPrefab
Stars Order In Layer	-10
Star Hour Start	18
Star Minute Start	30
Star Hour Finish	7
Star Minute Finish	0
Star Fade In And Out In Minutes	180
Parallax Scale	0.95

Spawn Stars: When true, this feature is switched on. Set to false if you do not intend to use this feature.

Stars Prefab: This is where you can add the object you wish to use for the stars - or use the default provided object.

Stars Order In Layer: Set the layer you wish stars to appear on. It is recommended to set this one higher than the background shades Order In Layer if in use. If you cannot see your Stars and it during the correct time, it may be due to this setting being lower than your background imagery. This is for use with an orthographic camera.

Star Hour/Minute Start/Finish & Fade: Using these settings you can determine in 24 hour time the start and end of the Stars cycle. In the above example, stars begin appearing at 6:30pm and disappear at 7am. You can also set how quickly the stars fade into view - to create the impression of stars slowly appearing. This fade out time is in minutes. In the above example, it is set to 180, or 3 hours - and will spend that time slowly fading into view.

Note: The start time is when the stars first begin to fade in. The end time is when the stars finish fading out. Please be mindful of this. In the above example, stars will begin to fade in at 6:30pm, but it won't be until 9:30pm (3 hours or 180 minutes later) that the stars will be fully in view. Further, the stars will start disappearing at 4am, three hours before 7am - so that they are fully gone by 7am.

Parallax Scale: Set the parallax scale of the stars. A value of 1 will follow the Main Camera exactly. A value of 0 will remain stationary. In the above example, a value of 0.95 will show the stars barely move, but will still slowly transition in the X direction as the camera moves.

Parallax is where objects further away move more slowly than objects up close. Imagine driving on a highway. The trees on the road zip on by, but the mountains in the distance barely move. In this sense, the trees have a very low parallax scale, and the mountains have a very high parallax scale.

Weather Generator

Weather object which controls all weather-related parameters. With this object, you can create new weather effects, apply existing weather effects, and update the wind settings. Weather Generator **requires** a Time and Background script to function.

Wind Settings

- Switch wind on and off easily
- Control and update the direction and magnitude of the wind
- Have weather elements directly affected by the power of wind

Weather Elements

- Spawn your own weather elements such as rain, snow, clouds or fog
- Set your own lightening object, or use the default provided prefab
- Control lightening frequency, and relevant collision layers
- Set lightening order in layer
- Create new, or add your own prefab weather elements
- Add collisions, splashes, whether wind affects it, and more quickly from the one object
- Spawn weather elements in inspector quickly

For more details on how to use each of these objects, please see the relevant next section.

Wind Settings

Control the wind settings that will effect weather elements/relevant particle effects. This is found on the Weather Object and is controlled by the following settings:

Wind Settings

Is Wind On

☐

Wind Direction And Magnitude

X

-4

Y

0

Is Wind On: Determines if the wind object is in place or not. This can be switched on and off during runtime.

Wind Direction and Magnitude: This is the 2D vector for the wind. This can determine the direction and strength of the wind. X goes left and right, -1 and 1 respectively. Y goes down and up, -1, and 1 respectively. Higher values increase the strength of the wind in that direction.

Note: Only weather elements/particle effects that have the wind settings enabled will be affected by this object. If you are building your own particle effect outside the weather object, be sure to enable External Forces and the spawned wind object to the particle effect.

The wind can be updated at runtime using the public function [ChangeWind](#).

Weather Elements

Weather elements can be customised and spawned through the Weather object. This covers a large variation in what can be done - from lightening to snow to fog or rain. It also supports spawning outside play mode for testing or manual adjustments. It is governed by these properties:

The screenshot shows the 'Weather Elements' inspector panel. It has a sidebar on the left with sections: 'Weather Elements', 'My Lightning', 'Weather Particle Effects', and 'Spawn Weather Elements'. The main area contains the following controls:

- Is Lightening On:** A checkbox that is currently unchecked.
- My Lightning:**
 - My Lightning Prefab:** A dropdown menu showing 'Flash'.
 - Lightening Frequency:** A slider with a value of 3.
 - Lightening Collision Layers:** A dropdown menu showing 'Environment'.
 - Lightening Order In Layer:** A text input field with the value 0.
- Weather Particle Effects:**
 - Size:** A text input field with the value 7.
 - Element 0 through Element 6:** A list of expandable sections, each with a right-pointing triangle icon.
- Spawn Weather Elements:**
 - Weather Element to Spawn:** A dropdown menu showing 'Light Rain'.
 - Spawn Weather Element: 'Light Rain':** A button at the bottom.

Is Lightening On: When true, lightening is on with the below settings. When false, lightening is switched off.

My Lightning Prefab: The object/particle effect used for lightening. Can be replaced or you can use the default provided lightening prefab. It is worth noting the prefab lightening prefab is just a flashing light cloud, that has a 50% chance of spawning a submitter lightening bolt (hence its name 'Flash'). Depending on the style of your lightening bolts, it may be appropriate to either replicate the default prefab design, or insert them separately to this object.

Lightening Frequency: Used to determine how frequently lightening occurs. The higher the number is, the more frequent lightening is.

Lightening Collision Layers: Used to set what can be hit by the lightening. When lightening strikes, the prefab version will emit a small particle effect on appropriate collision layers. You can select multiple layers in this list. For more information on collision layers, see the Unity documentation.

Lightening Order in Layer: Set the orthographic camera order in layer position for the lightening. This allows you to do things such as hide lightening behind clouds, or strike behind hills, but in front of mountains as required.

 Lightening can be modified during runtime using the public function [ChangeLightening](#)


The screenshot shows the 'Weather Particle Effects' inspector panel. It has a sidebar on the left with sections: 'Weather Particle Effects' and 'Element 0'. The main area contains the following controls:

- Weather Particle Effects:**
 - Size:** A text input field with the value 7.
- Element 0:**
 - Prefab Particle Effect (OPTIONAL: See Tooltip):**
 - Prefab Particle Effect:** A dropdown menu showing 'None (Game Object)'.
 - Spawn Location:** A dropdown menu showing 'Top'.
 - Custom Spawn Location:** Two text input fields for 'X' and 'Y', both with the value -1.


Weather Particle Effects: Weather particle effects is an array of elements that can be used to both design and spawn weather elements. There are many parameters to creating your own weather element, and you can see example cases below and in the provided sample scene.

The **Size** is the size of the array, which dictates how many different weather elements you have designed. Reducing this number will remove the lowest weather element, and increasing it will duplicate the last element for the value increased. Each element can be edited as required. The below is an example element:

Optional Prefab Particle Effect: If you have your own particle effect you have made and would like to use instead of filling out the below details, you can drag and drop the as a gameobject into this field. This will use the settings on the object and ignore any settings below, excluding spawn location.

 Note! The next section, Spawn Location - is a required field for all particle effects, whether using the optional prefab or building through the Weather object. If your weather object is not spawning in the desired location, it is likely due to this field not being set.

Required Spawn Location: This dropdown menu is required and dictates where the object will be spawned when created. If you would like to use a custom location, you can - by selecting custom in this drop down menu, and defining the location in the Vector2 'Custom Spawn Location' field.

 Coordinates (x, y) area based off (1, 1) equalling top right corner of the background, and (-1, -1) being the bottom left of the background. Values greater than 1 or less than -1 will move at a ratio equivalent to the background size in the relevant direction.

New Particle Effect Options	
Effect Name	Light Rain
Particle Lifetime	3
Particle Color	<div></div>
Particle Start Speed	0
Particle Min Max Size	X 0.1 Y 0.5
Gravity Modifier	3
Emission Rate	80
Shape Type	Single Sided Edge
Shape Size Multiplier	<div></div> 2
Change Size With Speed	<input checked="" type="checkbox"/>
Min Max X Size Change With Speed	X 0.1 Y 0.5
Min Max Y Size Change With Speed	X 1 Y 3
Sway Particle	<input type="checkbox"/>
X Sway Amount	<div></div> 0
Y Sway Amount	<div></div> 0
Enable Particle Collisions	<input checked="" type="checkbox"/>
Dampen On Collision	<div></div> 1
Bounce On Collision	<div></div> 0
Lifetime Loss On Collision	1
Collision Layers	Environment
Particle Collision Force	1
Prewarm Particle Effect	<input checked="" type="checkbox"/>

Effect Name: The name of the effect when it is spawned into the scene, and how it appears in the Inspector

Particle Lifetime: How long is each particle alive for. This is also used with the Emission Rate to determine the maximum number of particles able to be made.

Particle Color: What is the start color of the particle.

Particle Start Speed: What speed does the particle start with.

Particle Min Max Size: The starting size of the particle, as a random between these two constants

Gravity Modifier: The gravity acting on the object, as a multiplier based off the physics defined gravity. For more information on the physics defined gravity, see the Unity Documentation.

Emission Rate: The rate at which the particle effect generates. A higher emission creates a higher amount it appears. This is also used to calculate the maximum number of particles that can be generated, with particle lifetime.

Shape Type: The shape the particle effect spawns into. In general, only Single Sided Edge and Rectangle are used, however other shapes may be useful depending on your requirements. Single Sided Edge is essentially a 2D line - which works well for rain or snow. A rectangle the size of the background is useful for when spawning Fog and wanting it to appear mixed around on the Y axis.

Shape Size Multiplier: The size of the object is based of the size of the background. This multiplier is used to either increase or decrease the size as required. A value of 1 is the same size as the background.

Change Size with Speed: This setting determines the size based on the speed of the particle. The higher the speed, the more it moves towards the Y value. The lower the speed, the closer it skews towards the X value. This is applicable for both the x and y scaling values.

Sway Particle: Apply a sway effect to the particle. The X and Y sway amount determine how much sway each particle has as well. This is useful for creating snowfall like effects

Enable Particle Collisions: Determines whether this particle system can collide with other colliders

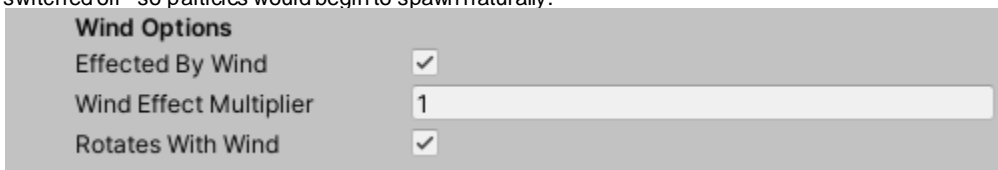
Dampen/Bounce On Collision: If particle collisions is enabled, these values will determine how much speed is either lost (dampen), or if a bounce effect occurs.

Lifetime Loss on Collision: How much life does the particle lose when colliding with another element.

Collision Layers: What layers can this particle collide with. Multiple layers can be selected. For more information regarding collision layers in 2D - see the Unity Documentation.

Particle Collision Force: How much force do the particles impact on the surface they collide with.

Prewarm Particle Effect: If switched on, the particle effect is prewarmed. That is, it will appear immediately as if it has already started to run. This is good for starting a scene with rain already falling. However, if you would like a particle effect like rain to slowly begin, this would be best switched off - so particles would begin to spawn naturally.



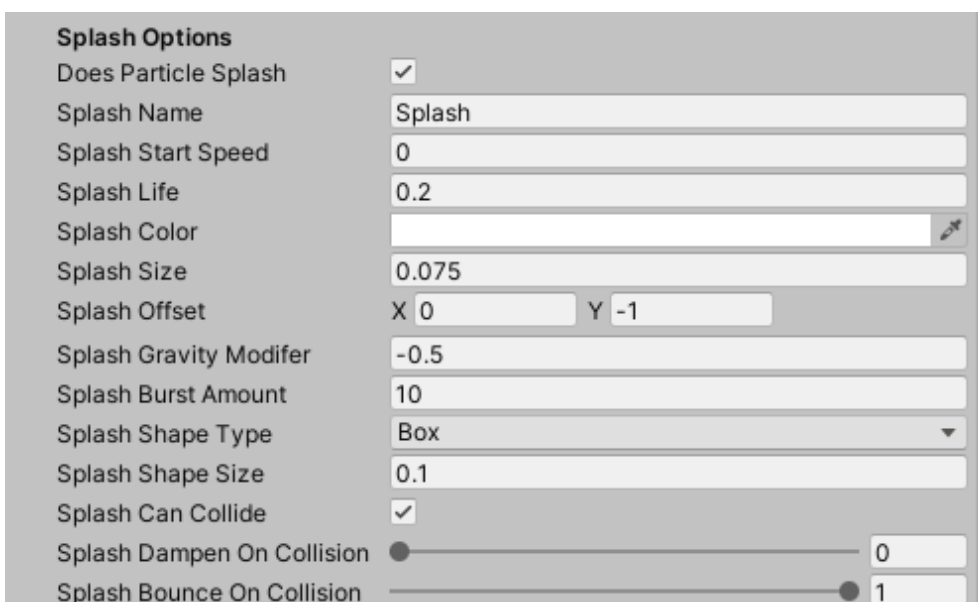
The screenshot shows the 'Wind Options' section of a particle system's inspector. It contains three settings: 'Effected By Wind' with a checked checkbox, 'Wind Effect Multiplier' with a text field containing the value '1', and 'Rotates With Wind' with a checked checkbox.

Wind Options: These three settings determine whether the particle effect is affected by the [Wind](#) object. If it is set to false, it will ignore the wind object.

Wind Effect Multiplier: This is how affected by wind strength the object is. A value of 1 is equal to the normal strength based on the wind.

Rotates with Wind: This determines if the wind rotates the object. For example, when this is switched on with rain, rain will turn to face the direction it is moved due to wind.

📌 When designing your own particle effect - if you would like it to rotate with wind, make sure it drawn to face downwards. The 'Rotates With Wind' parameter assumes the default rotation is straight down (like a rain drop). Any rotations based off wind are calculated accordingly.



The screenshot shows the 'Splash Options' section of a particle system's inspector. It contains the following settings: 'Does Particle Splash' (checked), 'Splash Name' (text field: 'Splash'), 'Splash Start Speed' (text field: '0'), 'Splash Life' (text field: '0.2'), 'Splash Color' (color picker), 'Splash Size' (text field: '0.075'), 'Splash Offset' (X: '0', Y: '-1'), 'Splash Gravity Modifier' (text field: '-0.5'), 'Splash Burst Amount' (text field: '10'), 'Splash Shape Type' (dropdown: 'Box'), 'Splash Shape Size' (text field: '0.1'), 'Splash Can Collide' (checked), 'Splash Dampen On Collision' (slider from 0 to 1, set to 0), and 'Splash Bounce On Collision' (slider from 0 to 1, set to 1).

Splash Options: This set of variables governs a subemitter created for particle effects. The subemitter acts as a splash generated on a collision. This will require a collisions to be enabled above. When **Does Particle Splash** is set to true, it will generate a subemitter. When false, it will not create a splash effect.

Splash Start Speed: Similar to the particle effect start speed - governs the speed when the splash particle effect is generated.

Splash Life: How long the splash particles exist for.

Splash Color: The starting color of the splash particle effect.

Splash Size: The size of the splash effect.

Splash Offset: Any offset in the X and Y position for the splash. This is useful for times like rain, where the ground is hit, but you would like the splash to appear slightly lower. In the above screenshot, it is set to -1, so as to make the splash appear a bit further down to where the collision occurred.

Splash Gravity Modifier: Similar to the particle effect gravity modifier. This is the multiplier of gravity affecting the splash effect - based off the Physics 2D Gravity value. For more information on Physics 2D Gravity, see the [Unity Documentation](#).

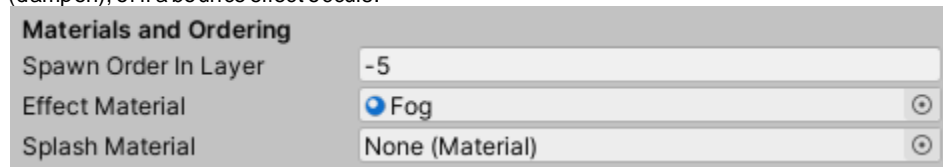
Splash Burst Amount: For each splash, how many particles are generated.

Splash Shape Type: The shape of the particle splash area.

Splash Shape Size: The size of the splash spawn shape's area

Splash Can Collide: Can the splash particles collide with other elements? The splash collision uses the same layers from the main particle effect collision layers.

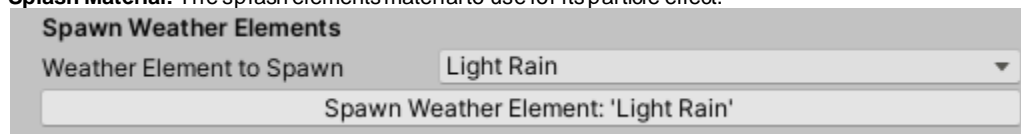
Splash Dampen/Bounce On Collision: If splash particle collisions is enabled, these values will determine how much speed is either lost (dampen), or if a bounce effect occurs.



Spawn Order In Layer: The order in layer for the sprite to appear with an orthographic camera.

Effect Material: The weather elements material to use for its particle effect.

Splash Material: The splash elements material to use for its particle effect.



Spawn Weather Elements: This is a weather element spawner for spawning elements either in game or out of game when setting up elements. Select the weather element to spawn from the drop down menu (it will retrieve this list from the above Weather Elements Array you have made), and select the button to spawn the element as it would if spawned in game. Spawning weather elements can also be done through the public function [SpawnWeatherElement](#).

Public Functions

```
void SetTime (bool isTimePaused, int changeTimeHour, int changeTimeMinutes, bool changeTime, float changeTimeRateInSeconds, bool changeTimeRate)
```

- Function to control time parameters
- **isTimePaused** - when true, pauses time. When false, time advances according to the **timeRate**.
- **changeTimeHour** and **changeTimeMinute** - changes the time to the set value given. The bool **changeTime** must also be set to true to change the time to those values. If it is false, **changeTimeHour** and **changeTimeMinutes** will be ignored.
- **changeTimeRateInSeconds** is the rate at which time moves in seconds. the bool **changeTimeRate** needs to be set to true, otherwise **changeTimeRateInSeconds** will be ignored. This value is in seconds and is related to real time. For example, a value of 1 will make the time parameter advance 1 second for every realtime second. A value of 3600 (1 hour in seconds) will advance time 1 hour for every realtime second.

```
void ChangeLightening (bool turnLighteningOn, float changeLighteningFrequency, bool useCurrentSettings)
```

- Function to control the lightening parameters
- **isLighteningOn** - when true, turns lightening on. When false, destroys the lightening object.
- **changeLighteningFrequency** - here a value can be set for the lightening frequency.
- **useCurrentSettings** - if you do not wish to change the lightening frequency, set this value to true to leave at whatever the current lightening frequency is.

```
void ChangeWind (bool turnWindOn, float xVector, float yVector, bool useCurrentSettings)
```

- Function to control the wind parameters
- **turnWindOn** - when true, turns wind on. When false, destroys wind object.
- **xVector** - set the xVector of the wind object.
- **yVector** - set the yVector of the wind object.
- **useCurrentSettings** - if you do not wish to change the wind x and y magnitude settings, set this value to true to leave it as the current settings.

```
SpawnWeatherElement (WeatherParticleEffect we)
```

- Function to spawn a weather element
- **we** - The weather element to spawn. You can get a list of available WeatherParticleEffect objects from the public array weatherParticleEffects.

Background Generator

2D procedural background generator for use in orthographic camera scenes. This script will generate random background within provided parameters. You can change the scale, colour, bumpiness, and parallax settings of the background, and either save for later use, or generate at runtime, or even randomize every time you play.

Repeat Build - Troubleshoot/Design Only

- Settings in this mode call the procedural generation of the background every frame. This is obviously not ideal, as it should only need to be run once. It exists as a way when designing backgrounds and playing with elements to tweak designs and see it in runtime. It can also be used to design the background before saving the mesh to avoid using the Background Generator in the final product at all.

Generation Settings

- Determine the size, style, and shape of your backgrounds.
- Set the color or gradient of the background.
- Randomise the generation.
- Apply an offset until you find the setup that works for you.

Looping

- Use looping to create an infinite repeating background.
- Control the level of smoothness between the looping elements.

Build Over Time

- Have your background procedurally generate at runtime - and watch background slowly fill in.
- Control the rate at which the background generates.

Outline Settings

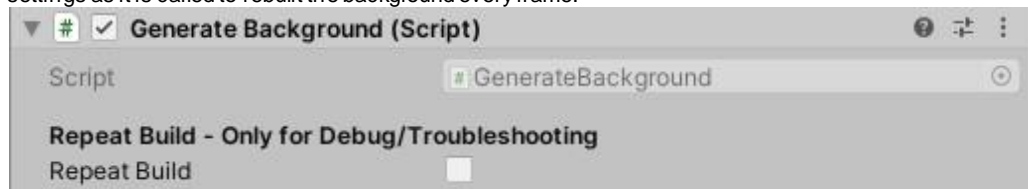
- Create an outline for your background.
- Color the outline as required.
- Set the thickness of the outline.

Camera Position Settings

- Set the orthographic camera the background is following (or leave blank to default to the main camera).
- Set the parallax level to create a sense of depth with your backgrounds.
- Set the order in layer to control where the background is rendered in a 2D environment.

Repeat Build - Troubleshooting/Design Only

Repeat build is a useful troubleshooting/design setting for building procedural backgrounds. It should **not** be used in most final builds/production settings as it is called to rebuild the background every frame.

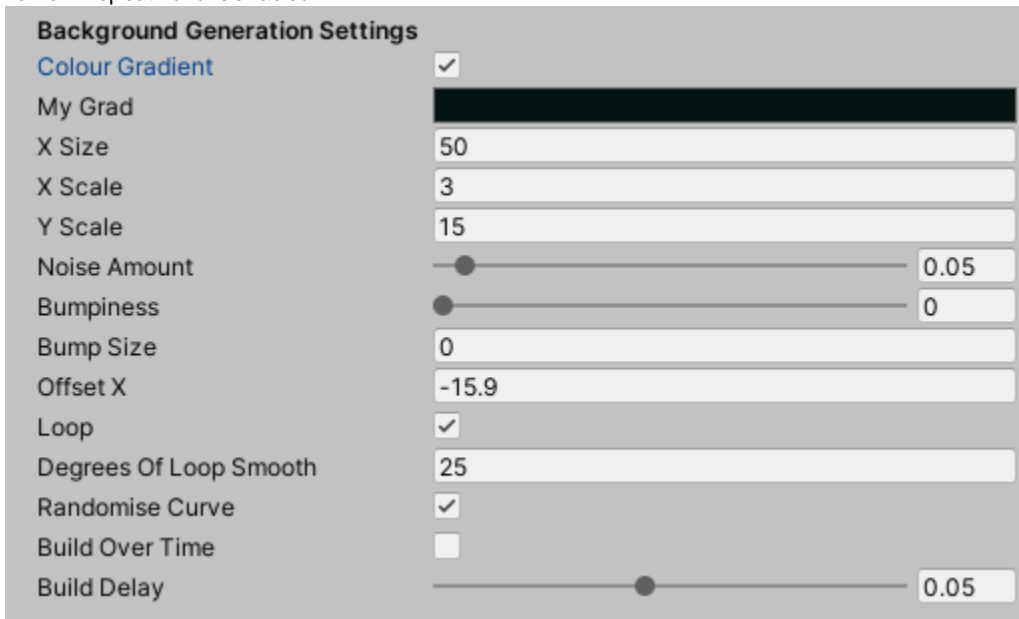


Repeat Build: Setting the value to true will cause the scene to rebuild the background mesh every frame. This is useful when trying to find the right settings for your background, but should not be used in final builds as it would waste CPU.

Note: Build Over Time cannot be enabled when repeat build is switched on

Generation Settings

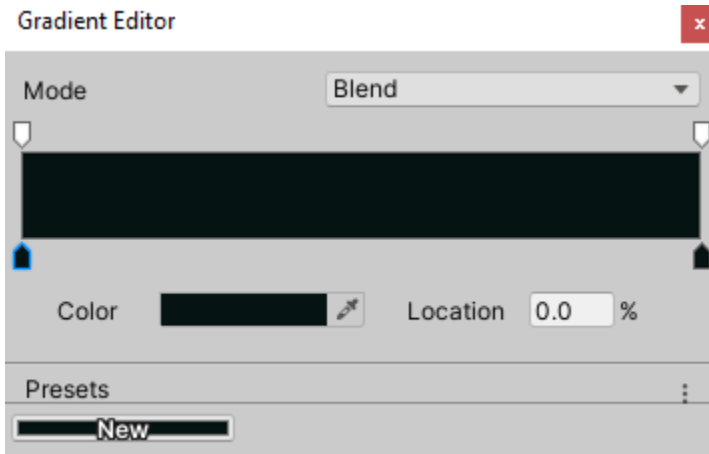
The generation settings control the style and look of your procedurally generated background. These generate on Start, or, can generate every frame if Repeat Build is enabled.



The 'Background Generation Settings' panel contains the following controls:

- Colour Gradient:** A checked checkbox.
- My Grad:** A color bar showing a gradient from black to white.
- X Size:** A text input field with the value 50.
- X Scale:** A text input field with the value 3.
- Y Scale:** A text input field with the value 15.
- Noise Amount:** A slider ranging from 0 to 0.05, currently set at 0.05.
- Bumpiness:** A slider ranging from 0 to 0, currently set at 0.
- Bump Size:** A text input field with the value 0.
- Offset X:** A text input field with the value -15.9.
- Loop:** A checked checkbox.
- Degrees Of Loop Smooth:** A text input field with the value 25.
- Randomise Curve:** A checked checkbox.
- Build Over Time:** An unchecked checkbox.
- Build Delay:** A slider ranging from 0 to 0.05, currently set at 0.05.

Color Gradient: When true, will color in the generated background according to My Grad. My Grad can be a single color, or a gradient, as designed in the gradient settings. It will treat the left most side as the bottom of the background, and the right most side as the top of the background.



The 'Gradient Editor' panel includes the following elements:

- Mode:** A dropdown menu set to 'Blend'.
- Color Bar:** A horizontal bar showing a gradient from blue to black.
- Color:** A color picker set to black.
- Location:** A text input field with the value 0.0 %.
- Presets:** A list of preset gradients, with 'New' selected.

For more information on the Gradient Editor, see the Unity Documentation.

X Size: Sets how many units wide the generated background is.

X Scale: Sets how distant each section of the generated background is. If this number is large, the difference between each point is significant, and it will create a very straight line look. If this number is small, it will create a short distance between each point resulting in a smoother change.

Note: Both X Size and X Scale contribute towards the overall width of the background. Make sure to compensate for your desired X Scale by increasing/decreasing the X Size as required.

Y Scale: This determines how tall your background is.

Noise Amount: This is where you can control the primary curve. Set a value between 0 and 0.999, where a low number results in smooth looking hills, and a high number increases the chaotic-ness of the curve, resulting in rocky looking terrain.

Bumpiness: This is where you can control the bumpiness. It adds a second level of 'noise' to the generated field to increase the bumpiness without affecting the overall shape. This is useful creating 'bumps' on otherwise smooth hills for a more natural look. A value of 0 is no bumps, and like noise, it goes up to 0.999.

Bump Size: This controls the how big the bumps generated are when in use.

Offset X: A value to offset the generated curve. The curve procedurally generates using Perlin Noise, and so an offset will allow you to select different places to use for curve generation. Use this to move your background generated spot around until you find your desired landscape.

Randomise Curve: This is only called on the first frame, and will randomise an Offset while keeping the rest of your conditions. Use this to either create a new background everytime you lead the scene, or for searching random spots to use for your background. This will not change your other settings - but will randomise a curve based off them.

Looping

Create infinite loops for procedural backgrounds. With this setting, the background modifies the end of its generated mesh to line up with the start of the mesh, so it can be seamlessly looped.

Loop	<input checked="" type="checkbox"/>
Degrees Of Loop Smooth	<input type="text" value="25"/>

Loop: If enabled, this will loop the background.

Degrees Of Loop Smooth: This is how many units (based off the X Size) the background uses when re-shaping itself to ensure it loops up with the beginning. This must be less than half the X Size.



Loop cannot be enabled during runtime. This is a setting that is normally saved until after you have already designed the general mesh of your background.

Build Over Time

Building the mesh over time stops the mesh from immediately generating, but shows the process of building as it generates each section step by step.



The image shows a UI control panel with two settings. The first setting, 'Build Over Time', is a checkbox that is currently unchecked. The second setting, 'Build Delay', is a slider control with a circular knob positioned in the middle. To the right of the slider is a numerical input field containing the value '0.05'.

Build Over Time: When enabled, the mesh will build over time at the start of the scene. Otherwise, it will build immediately.

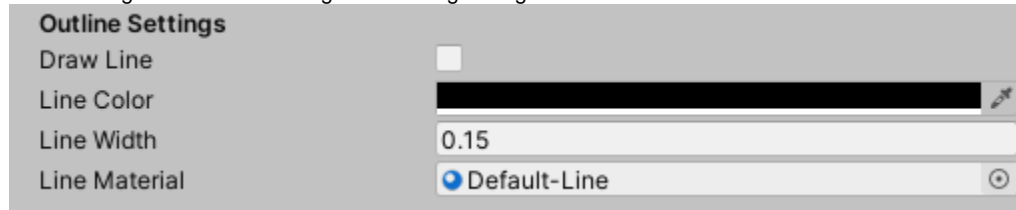
Build Delay: This is the delay between each step as it builds. A lower delay makes for a faster build, and a higher delay makes for a slower build.



Build Over Time cannot be used when Repeat Build is enabled. Repeat Build relies on immediately building the mesh.

Outline Settings

Outline the generated mesh using the following settings.



The screenshot shows a settings panel titled "Outline Settings". It contains four configuration options:

- Draw Line:** A checkbox that is currently unchecked.
- Line Color:** A color selection bar showing black, with a small icon to its right.
- Line Width:** A text input field containing the value "0.15".
- Line Material:** A dropdown menu showing "Default-Line" as the selected option, with a circular icon to its right.

Draw Line: When enabled, the mesh will draw a line around itself when complete.

Line Color: Set the colour of the line.

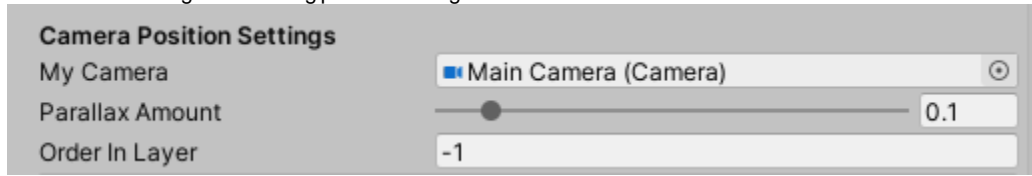
Line Width: Set the width of the line.

Line Material: Set the material used by the like (Default is Default-Line)

i This setting can be updated with repeat build, but once built will remain unless you specifically delete the created Outline gameobject.

Camera Position Settings

The camera settings for creating parallax backgrounds.



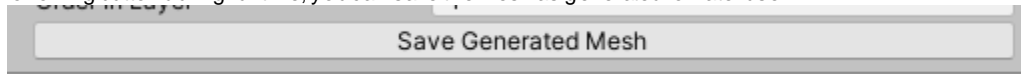
My Camera: What camera does the background look at to follow. If blank, it will set itself to the Main Camera.

Parallax Amount: Set the parallax amount of the background. A value of 1 will follow the Main Camera exactly. A value of 0 will remain stationary. In the above example, a value of 0.1 will show the background stand almost still, but follow the background slightly, to give a sense of depth as the scene moves forward/backwards on the X axis.

Order In Layer: Set the layer of the generated background mesh appear on. This is for use with an orthographic camera. The layer controls the rendering order in 2D environments. For more information about Order In Layer, see the Unity Documentation.


Saving A Generated Mesh


After generating a mesh and getting the settings correct, you may not wish to call the Generate Background Script again. By selecting the following button during runtime, you can save the mesh as generated for later use.



After selecting this button, you will be prompted with where you wish to save the object.

Once you have saved the mesh, you can now apply it to the Mesh Filter, and delete the 'Generate Background' component from the object.

 Saving the mesh is good practice to ensure things appear as you want. It also allows you to view your mesh in the scene view without having to hit Play.

 If you want your saved mesh to loop, you will still need to add the script/component 'Parallax Object'. When making this script, you will likely need to know the 'Total X Size' of the mesh, and it is easiest to get this value through generating the mesh and writing down the value before saving. For more information, see [Parallax Object](#).

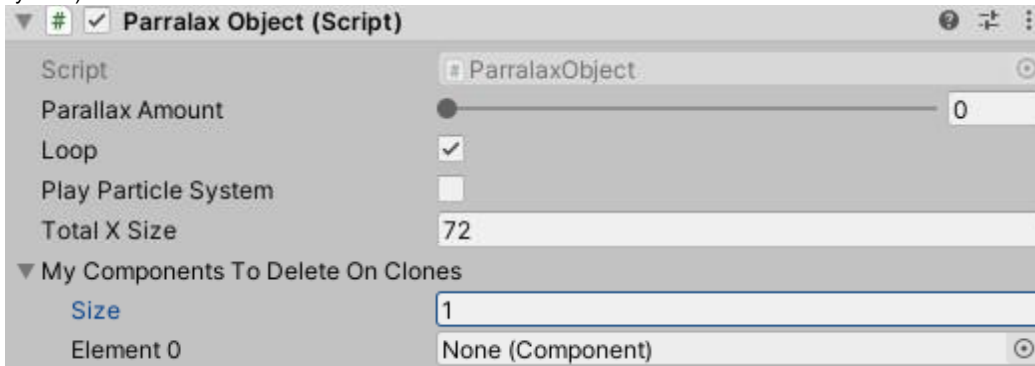
Parallax Object

Parallax Object (misspelt to ParralaxObject) is a simple script to loop a background element. This can be used on any object you wish to loop according to the camera.

When generating stars, or procedural backgrounds, this component is added to the elements to create both looping and parallax scaling. This component can work on BYO objects as well. It functions by generating rendered clones of the object, and removing any unnecessary scripts from the clones. It then tracks the position of the camera, its parallax scale, and the position of its clones, and loops as required.

Settings

Use the following settings on either procedurally generated backgrounds, or your own objects you wish to loop/parallax (such as a particle system).




Parallax Amount: Set the parallax amount of the object. A value of 1 will follow the Main Camera exactly. A value of 0 will remain stationary. This is used to give a sense of depth to objects. Think about driving on the highway. The trees by the road zip by fast, but the mountains in the distance go slowly. In this case, the trees have a very low parallax amount, but the mountain has a very high parallax amount. Using this, you can create depth in your backgrounds.

Loop: Is looping enabled? If this is false, it will not run looping scripts. If it is true, it will generate clones and run looping scripts.

Play Particle System: For ParticleSystem specific objects - this needs to be enabled to play particle systems on clones as required.

Total X Size: This is the manually set Total X Size. For procedurally generated backgrounds, this value is calculated and passed to the Parallax Object component - however if you would like to use this object for other BYO elements, you will need to get the width of the object and store it here.

My Components To Delete On Clones: The component works by generating clones of the object and placing them before and after each other - and then seamlessly moving between them to create a perfect loop. When generating clones, the script automatically deletes the 'Parralax Object' script from the clones, however any other scripts or components you do not wish to be on the clones can be removed by dragging them in here. This will not remove the scripts/components from the main object.

 Be careful when using this script if you create it programmatically. You can easily create a never-ending loop of creating clones if you do not make sure you 'myComponentsToDeleteOnClones' is properly set. This will crash Unity.

Contact/Troubleshooting

For any queries, concerns, bugs or suggestions, please email traffycontact@gmail.com

Thank you for your support.