GENA PRO

By Procedural Worlds

GeNa Pro is a sophisticated spawning system that enables you to rapidly and intuitively populate your scenes with assets.

Contents

About Procedural Worlds

Powerful, simple, beautiful. Friendly tools, gorgeous games!

Procedural Worlds empowers artists and developers to bring their vision to life by making it easy to create beautiful worlds. Leverage the latest procedural generation techniques to take the pain out of creating stunning environments and focus on creating amazing games.

NEW – Canopy community site

Canopy is our new community site for all Procedural Worlds Tools with support forums, knowledge base library, tutorials, and much more.
Register with Canopy to receive FREE stamp packs for Gaia and get the best out of Ambient Sounds.

The only end to end environmental generation and delivery suite:

Gaia Pro 2021 - A world generation system for creating, texturing, planting and populating scenes from low poly mobile, VR and through to high end desktop.

GeNa Pro - A sophisticated localised level design tool that augments Gaia's broad-brush strokes, by working intuitively to give fine grained control.

SECTR - A suite of performance-enhancing tools that enable open world streaming, massive mobile games and includes the latest techniques in audio occlusion and propagation.

Ambient Sounds - Lets you configure music and sounds to create a unique atmosphere for each region in your game, which can react to changes in your gameplay instantly.

Pegasus - A cut scene and fly through creator that makes it easy to show off gorgeous environments and drive characters through scenes with localised avoidance and Mecanim animation support.

Spawner Packs – You can save time by using our pre-configured Procedural Worlds Spawner packs (PWS). The packs contain configurations for our tools Gaia and GeNa, and are designed to work with popular asset packs from the Unity Asset Store. Currently available:

PWS – POLYGON Fantasy Kingdom - Spawner Pack
PWS – POLYGON Nature - Spawner Pack

Micro Biomes - Let us inspire you with our new Micro Biome series where we put together groups of matching assets that cover one specific aspect of environmental design and release them in targeted high quality packs:
Micro Biomes – Fields of Color

Introduction

Thank you for purchasing GeNa Pro!

Born from years of commercial work, GeNa Pro can populate and shape terrains, create villages and other interesting and complex man made structures, and can also create roads and rivers.

GeNa Pro is at ease at runtime as it is at design time and was designed with API control and extensibility in mind, so you can extend GeNa Pro and embed it into your own solutions.

GeNa Pro is integrated with Gaia Pro so that it is simple and fast to create environments that 'just work'. It also comes with sample spawner packs for Gaia Pro 2021 and the free Flooded Grounds asset, so you can be up and running around in your own levels within minutes.

Concepts

Based on the Wagiman Aboriginal word "Ge-Na" means "to put" or "to plant".

GeNa enables you to shape and texture your terrains, and then populate them with resources including terrain textures, terrain details (grass), terrain trees, rivers, roads, prefabs, and sophisticated procedurally generated structures.

GeNa does this with **Spawners**, **Spawner Decorators**, **Splines** and **Spline Extensions**.

Spawners add content into your scene at design time or run time. They can be saved as prefabs and reused at any time in the future.

Spawners use **Spawner Palettes** to maintain references to the content you are spawning into your scene.

Spawners have a sophisticated Spawn Criteria system that influence where content can be added to the scene. The impact of spawn criteria can be visualized by pressing the shift button.

Spawning can take place in several ways:

- ***Single Spawn*** – CTRL + Left click single spawn.
- ***Painted Spawn*** – CTRL + Left click and drag – spawns based on flow rate.
- ***Global Spawn*** – SHIFT + CTRL + Left mouse – spawns across the entire terrain.
- ***Spline Based Spawn*** – Spawner is called by the spline system along spline.
- ***API Based Spawn*** – Spawner is called via API.

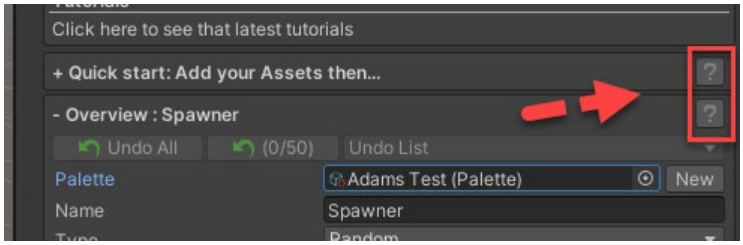
The Spawner system can be extended by the **Spawner Decorator** system. Decorators are scripts you add to game objects before ingesting them into a Spawner that influence the way the spawner operates when spawning.

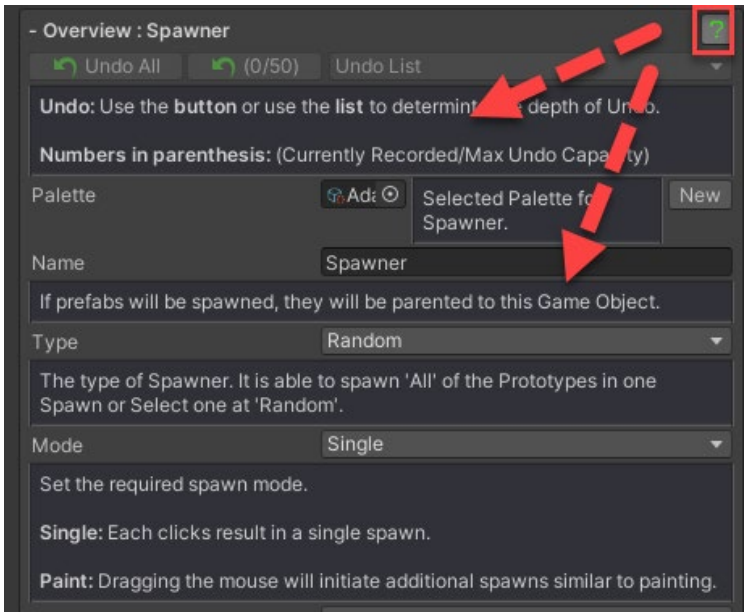
For example, a Terrain Decorator can flatten the terrain under a building when it is spawned into the scene. The Decorator system is extensible.

Splines and Spline Extensions either directly manipulate your environment as is the case of Rivers and Roads and can also apply Spawners along the spline. The Spline Extension system is extensible.

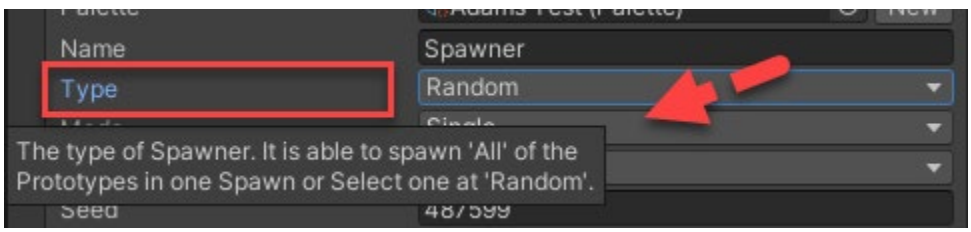
In Place Help

To understand the intended usage of the settings in a panel use the in-place help system. Toggle in-place help by pressing the "?" button on the panel. *NOTE:* In Place help will always be more up to date that the written documentation.





You can also select and hover over the title of a control to get a quick description of what the impact of the field is.



Content Pack Installation & Usage

GeNa Pro has been supplied with a significant amount of sample content.

This is a great place to start to get immediate value from GeNa Pro. It can be used to populate your scene with content, and you can also get a sense of the different ways you can configure GeNa by looking at the way the prefabs have been configured.

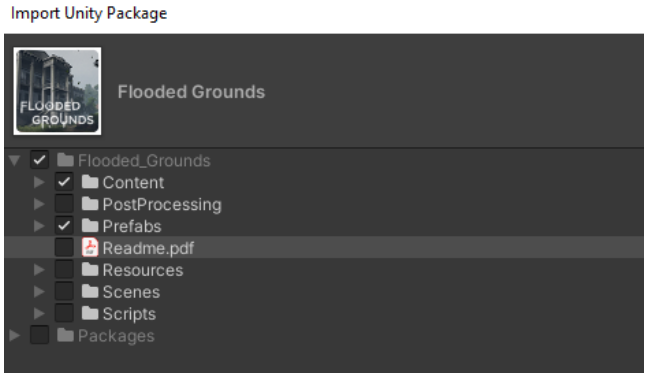
Keep an eye on the asset store, as we will be publishing additional content packs out in the future.

Gaia Pro Content Pack:

1. Install Gaia Pro 2021, open the Gaia Manager and follow the initial configuration process.
2. Double click on the Gaia Pro Content Pack package in the Procedural Worlds \ Asset Samples \ Content Packs directory to install the content pack.
3. Go to the Procedural Worlds \ Content Packs \ Gaia Pro \ Spawners directory and drag and drop the various Spawner prefabs into your scene.
4. Select the spawner prefab in your scene hierarch and then use it like usual to spawn preconfigured Gaia Pro content into your scene.

Flooded Grounds Content Pack:

1. Install the free Flooded Grounds pack from the Asset Store at Flooded Grounds | 3D Environments | Unity Asset Store
2. **NOTE:** Include only the Content and Prefabs folders, as the other folders are out of date and will most likely break your project. To stop the other folders from being included uncheck them before hitting import.



3. If you are using URP or HDRP, then you will need to follow the normal Unity Pipeline Content Upgrade Process. Can be found here: Unity Legacy to High Definition Render Pipeline conversion tutorial.
4. Go to the Procedural Worlds \ Content Packs \ Flooded Ground \ Spawners directory and drag and drop the various Spawner prefabs into your scene.
5. Select the spawner prefab in your scene hierarchy and then use it like usual to spawn preconfigured Fallen Grounds content into your scene.

API Reference & Script Examples

GeNa Pro script API's documentation can be found in the Documentation folder, in the GeNa Pro API Reference document.

Examples of these API's in user can be found in the Asset Samples / Script Examples / Scenes folder. Open the relevant sample scene and see how to control GeNa Pro via Script.

Tutorials, Chat, Ticketed Support

To help you take advantage of GeNa Pro we have created an awesome support network:

Discord: https://discord.gg/TggjONN

Tutorials: https://canopy.procedural-worlds.com/library/tools/gena-pro/37_tutorials/

Support: https://www.procedural-worlds.com/support/

GeNa Spawner

Spawners add content into your scene at design time or run time. They can be saved as prefabs and reused at any time in the future.

Spawners use **Spawner Palettes** to maintain references to the content you are spawning into your scene.

Spawners have a sophisticated Spawn Criteria system that influence where content can be added to the scene. The impact of spawn criteria can be visualized by pressing the shift button.

Spawning can take place in several ways:

- ***Single Spawn*** – CTRL + Left click single spawn.
- ***Painted Spawn*** – CTRL + Left click and drag – spawns based on flow rate.
- ***Global Spawn*** – SHIFT + CTRL + Left mouse – spawns across the entire terrain.
- ***Spline Based Spawn*** – Spawner is called by the spline system along spline.
- ***API Based Spawn*** – Spawner is called via API.

The Spawner system can be extended by the **Spawner Decorator** system. Decorators are scripts you add to game objects before ingesting them into a Spawner that influence the way the spawner operates when spawning.

For example, a Terrain Decorator can flatten the terrain under a building when it is spawned into the scene. The Decorator system is extensible.

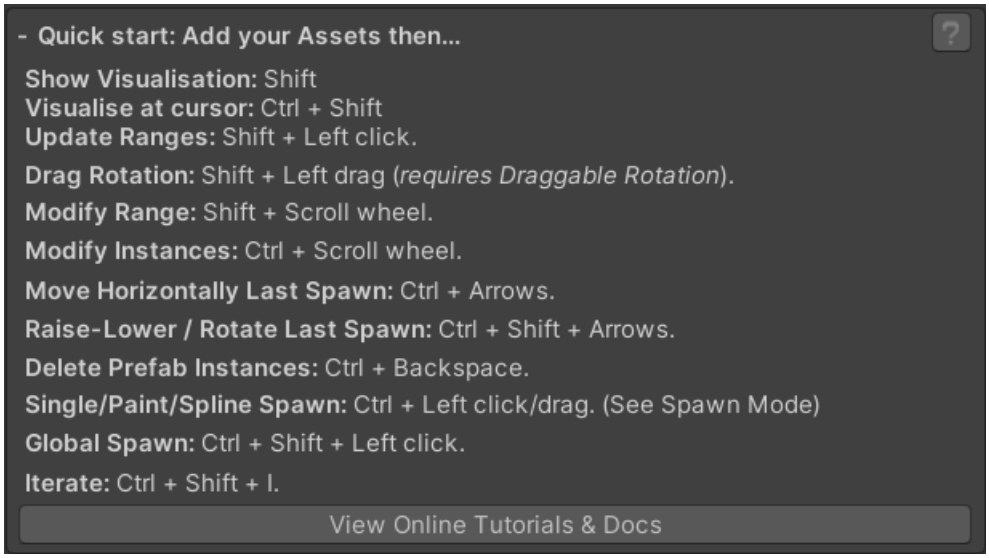
Interface

The spawner interface is hierarchical and made up of panels that should be read from the top down. The settings made in higher-level panels are influenced and modified by the settings in lower level panels, all the way down to the settings of an individual resource.

For example, the Rotation Type and Scale selected by GeNa in its Placement Criteria panel will be further modified by its prototype as it instantiated during spawning into the scene.

The idea is to refine the spawners operation by incrementally changing its settings, and when you have them right you can save your spawners as prefabs to preserve their settings for re-use in the future.

Quick Start Panel



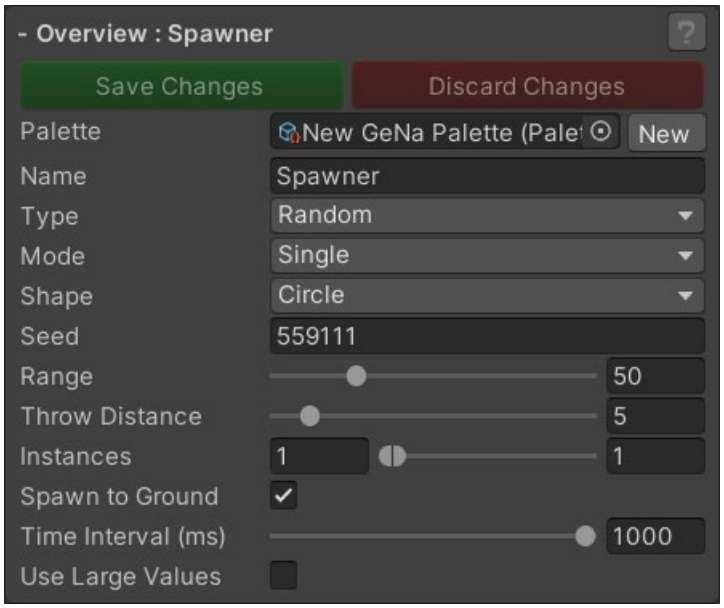
The Quick Start panel shows the mouse and keyboard controls used to control the spawner.

It is provided for convenience and can be opened and closed by clicking on the title.

For more information on the keyboard and mouse controls please see the Mouse & Keyboard controls section later in this document.

Overview Panel

The overview section shows the primary spawner controls.



Palette: The spawner palette. A scriptable object with contains references to all the assets used by a GeNa Spawner. It is good practice to create one palette for each of the asset packs you are using in your project and then share it among multiple GeNa spawners. Click New button to create a new palette.

Name: The name of the spawner.

Type: The type of Spawner. Spawn *All* of the Prototypes in one spawn or select a single Prototype at *Random*.

Mode: The spawn mode of the spawner. It can act as a *Single* click spawner or as a *Paint* spawner where you click and drag to control the spawning flow. Different options will show depending on what mode the spawner is in.

Flow Rate (Paint Mode): The distance between spawns when Paint Mode is selected. As you hit Ctrl + Left Click, and then drag while holding down Ctrl, a new spawn will be initiated after you have dragged the mouse beyond this distance.

Shape: The shape of the spawn – circle or square.

Seed: The seed used by the spawner is used to seed and initialize the random number generator that the spawner uses to generate output. This means that a spawner is capable of deterministic outputs, which can be great for networked games. Resetting the spawner via API with set the spawner back to its initial seeded state.

Range: The maximum range of the spawner in meters. Prototypes will only be spawned in this range around the central spawn location.

Throw Distance: The distance that will be used by the spawn algorithms to place spawned assets into your scene. I's usage will vary from algorithm to algorithm. Set this to zero when in Paint mode to have your spawned objects accurately follow your mouse.

Instances: The number of prototype instances that will be spawned per spawn. The actual instances spawned will vary between minimum and maximum values. This allows successive iterations to spawn different numbers of instances which is a great way of introducing variability into your spawns.

Sometimes the spawn criteria you choose will not spawn any instances regardless of these settings. The reason for this is that GeNa has a fail-safe to stop it from going into an infinite loop when your spawner settings mean that no content can be spawned. For example, if you are using bounds-based collision checking, and have a very small throw distance, then new objects will not be spawned far enough away from the original, and collisions with the original would stop new instances from being created.

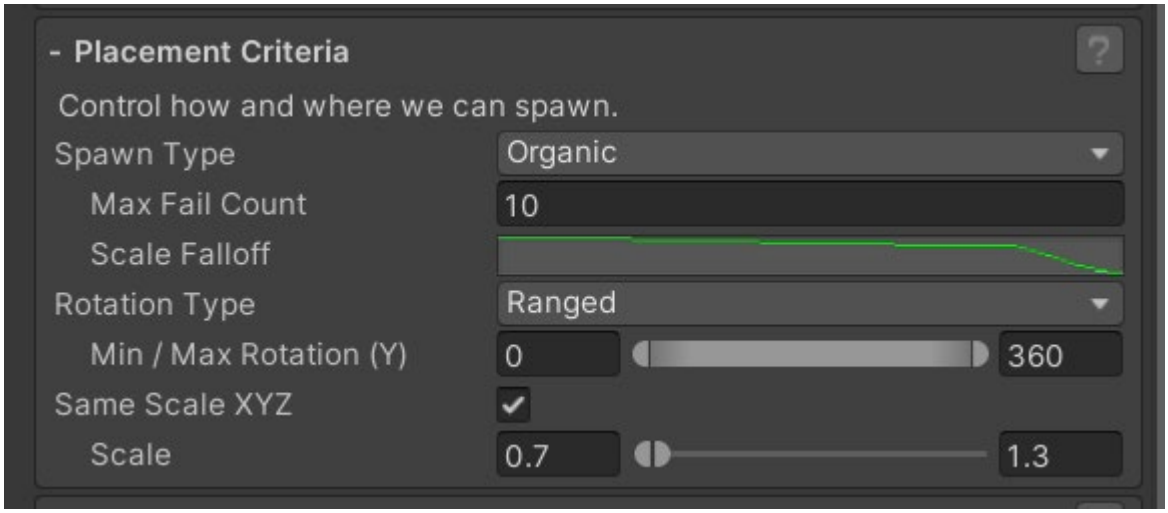
Spawn to Ground: Spawning happens relative to the hit point on the terrain or mesh that has a collider.

Spawn Time Interval (ms): The amount of time that each internal iteration of the spawner takes in milliseconds. Set this a smaller value to run the spawn process in smaller time increments over a larger time frame, and hence to have a smaller impact on performance at runtime for large spawns. Spawning is done via co-routine.

Use Large Values: Will enable you to set large values for spawn settings.

Placement Criteria Panel

This section controls the algorithms used to select how and where Prototypes are placed into your scene. It will change based on how the spawner is configured.



Spawn Type: The algorithm used to control where new prototype instances are spawned. All distances are all related to the Throw Distance set in the Overview panel.

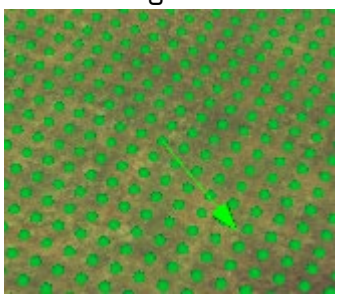
- ***Centered***: All spawn attempts will be centred from where you clicked up to the Throw Distance.
- ***Every***: Spawns will be attempted at every location in the Spawn Range. Locations will start at the edge of the spawn range and then be incremented by the Throw Distance, plus or minus the Throw Distance * Jitter Strength. Set Jitter Strength to zero for straight lines, and to one for random lines.
- ***Organic***: Spawns will grow out from the spawn location an organic matter. Imagine planting a seed and then watching it grow and cast other seeds which also grow. Each internal spawner iteration will use each seed as the base to cast a new seed, however, each seed maintains a failure count, and once a speed tries and fails to spawn around itself beyond that count, it will no longer attempt to cast new seeds. The impact of this is that spawns at the edge of the group are more likely to succeed than spawns in the middle and the group will spread out in more of an organic pattern. Additionally, the **Scale Falloff** curve influences the scale of each item spawned. The scale is applied over the Instances spawned from left to right up to the instances setting in the Spawner Overview. A great example way to use this is to spawn a large number of trees. The last trees to spawn (which are typically at the edge due to the failure count) will be smaller, and this simulates new growth nicely.
- ***Last Spawn***: Spawns will be attempted from up to the Throw Distance from last instance spawned.

Jitter Strength (*Every*): Only shown when Every Location has been chosen as the Spawn Type. Controls how much jitter is injected into the selection of the next location. A value of zero will generate straight lines at intervals based on the Throw Distance (defined in the

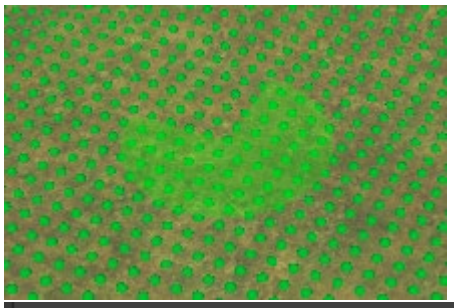
Overview setting), and a value of one will randomise the next location over the throw distance.

Rotation Type: Controls how the Prototype is rotated in the Y axis when it is spawned – i.e., which direction its points in. Individual resources within a prototype will be further offset by adding their own offset to the overall rotation that is applied so that they maintain their rotation relative to the parent prototype.

- *Fixed*: Rotation will be fixed at the angle specified. You can additionally check the draggable rotation check box and click Shift + Left click and drag to change the rotation angle with the mouse.



- *Ranged*: Rotation will be randomised between the Min and Max rotation settings. This is visualised as an arc when there is a range:



- *Last Spawn Center*: Spawned objects will be rotated from the centre of the last spawned object to the current spawn location.
- *Last Spawn Closest*: Spawned objects will be rotated from the closest point of the last spawned object to the current spawn location. This will get better results with things like fences.

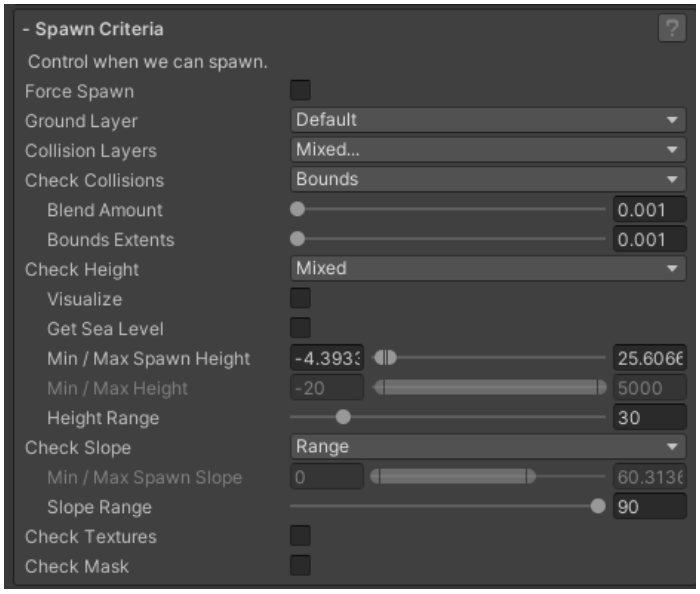
Same Scale XYZ: When selected will use the same scale in the x, y, and z axis, otherwise will use the specific scale overrides.

Min & Max Scale: The object spawned will be randomly scaled in this range.
Min & Max Scale X, Y, Z: The object spawned will be randomly scaled in these ranges.

NOTE: These scales are further influenced by the scales set per prototype. The final scale of the instantiated object will be the This Scale * Prototype Scale.

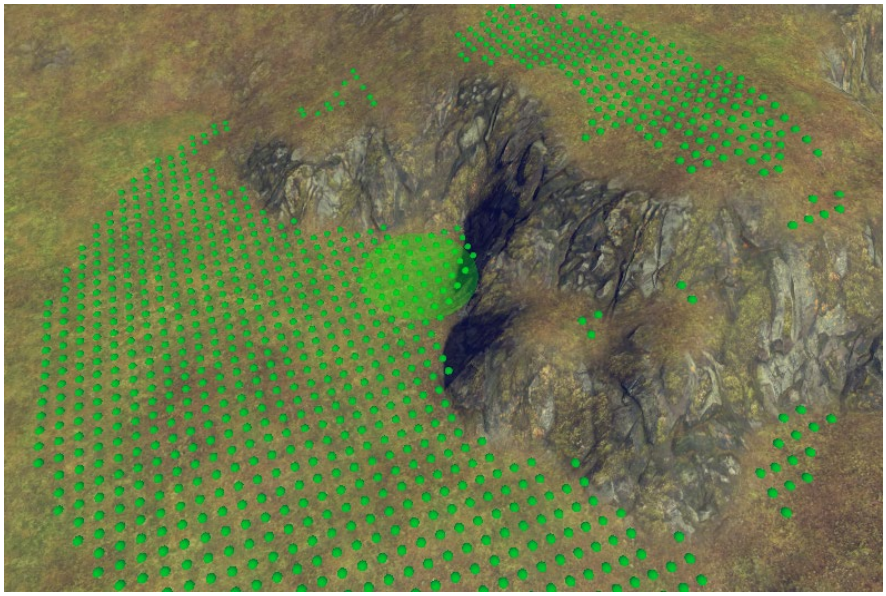
Spawn Criteria Panel & Spawner Visualization

This section controls when we can spawn.



The spawner uses the characteristics of the location you shift clicked on a terrain or mesh to find similar locations on that terrain or mesh, within the ranges set in your criteria.

For example, if you click at a height of say 50m, and have a height range of 20m, then it will select for all heights 10m either side of where you clicked, for an overall range of 20m.



The spawner helps you to understand the significance of your spawn criteria via the visualiser. Hit Shift + Left Mouse Button to set a location as your visualisation source and update the spawner ranges / texture settings based on that location.

You can modify your spawn criteria and see its impact in real time at that location by holding the Shift button while also changing the spawn criteria. This is a great way to fine tune it.

To see how a different location would be affected by the same spawn criteria, hit Ctrl + Shift and move the mouse around.

Force Spawn: Will always force a Prototype to spawn at the location you click on.

Ground Layer: Use this to control which layers will generate collisions when setting the ground object for your spawn. The ground object is then used as the ground for object placement when spawning.

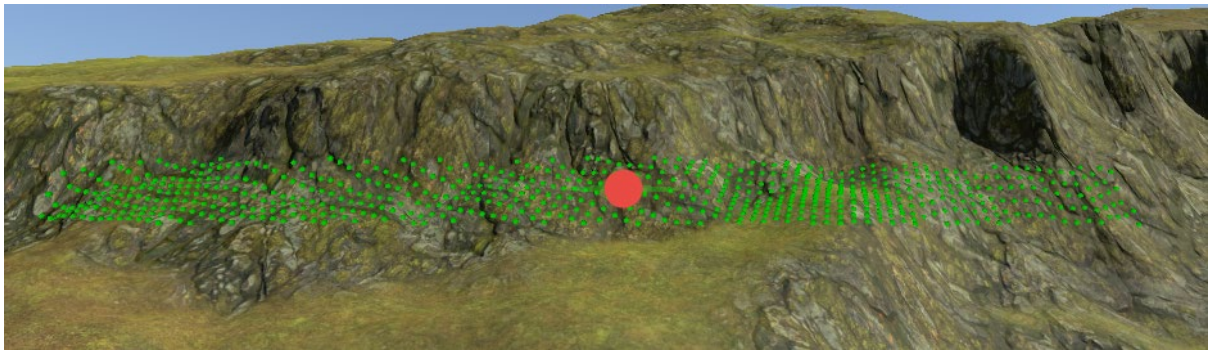
Check Collisions: Controls whether the spawner will spawn its Prototype in a space that is clear of collisions or not.

- *None*: Don't check for collisions.
- *Point*: Only check at the point the spawn was initiated. Good for grouping things closely together.
- *Bounds*: Check the entire volume (bounds) of the object to be spawned.
    - *Blend Amount*: Blends the surrounding object bounds together into a smoother averaged shape.
    - *Bounds Extents*: This is the distance beyond the edge of the bounds of the object we are placing that is checked against the blended surrounding.

Collision Layers: Use this to control which layers are tested when Point and Bounds collision checking is selected.

Check Height: Determines whether height is a factor in choosing whether the spawner will spawn a Prototype.

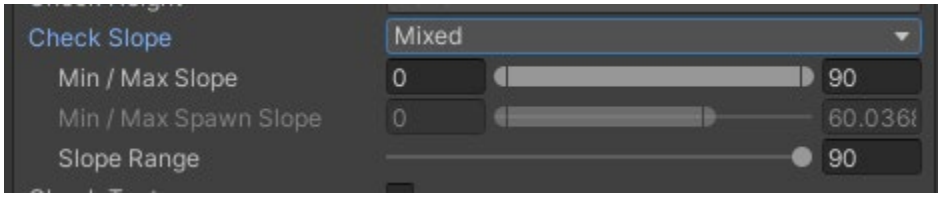
- ***None***: Do not use height check.
- ***Range***: Spawns only within the height range bisected by the height of the point where the visualisation or spawn was initiated i.e., where you shift or ctrl left clicked on the terrain or mesh.



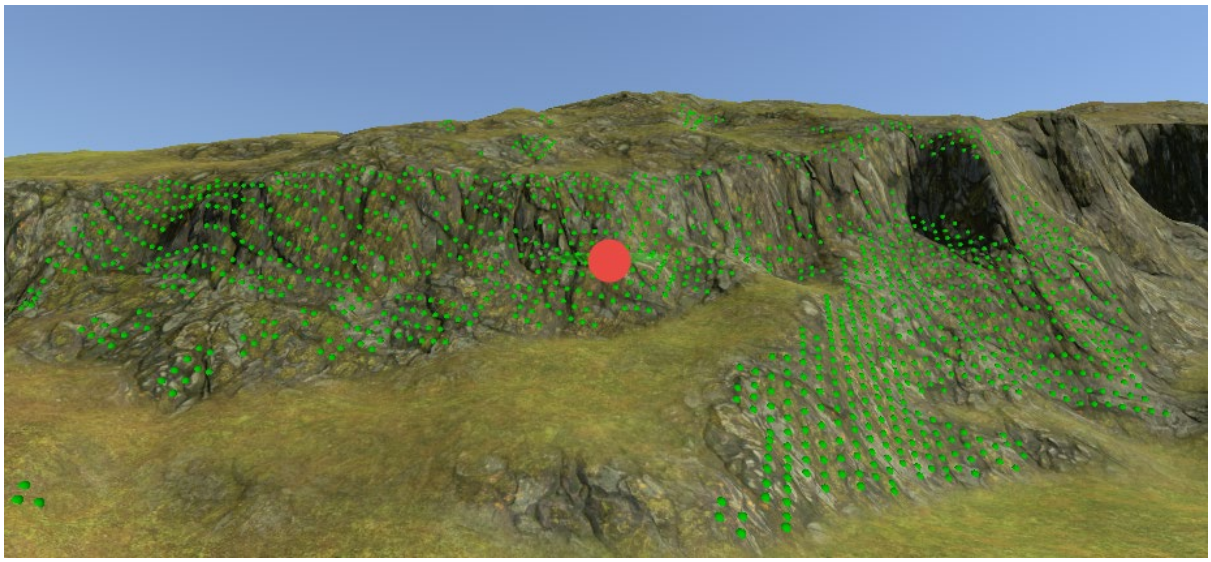
In this example, we have chosen a height range of 6 meters, a minimum height of 50 meters, and you can see approximately where on the terrain the shift click occurred. The visualiser shows the range of possible locations that it will consider for spawning.

- ***Min Max***: The minimum and maximum heights at which something will be spawned.
- ***Mixed***: The minimum and maximum heights at which something will be spawned, then further restricted by height range.
- ***Get Sea Level***: If Gaia is installed, will grab the sea level from Gaia and influence the minimum height.
- ***Min Max Spawn Height***: The minimum and maximum heights at which something will be spawned.
- ***Height Range***: Spawns only within the height range bisected by the height of the point where the visualisation or spawn was initiated i.e., where you shift or ctrl left clicked on the terrain or mesh.
- ***Visualize:*** Show a visualization if the impact of the heights.

Check Slope: Determines whether slope is a factor in choosing whether the spawner will spawn a Prototype.



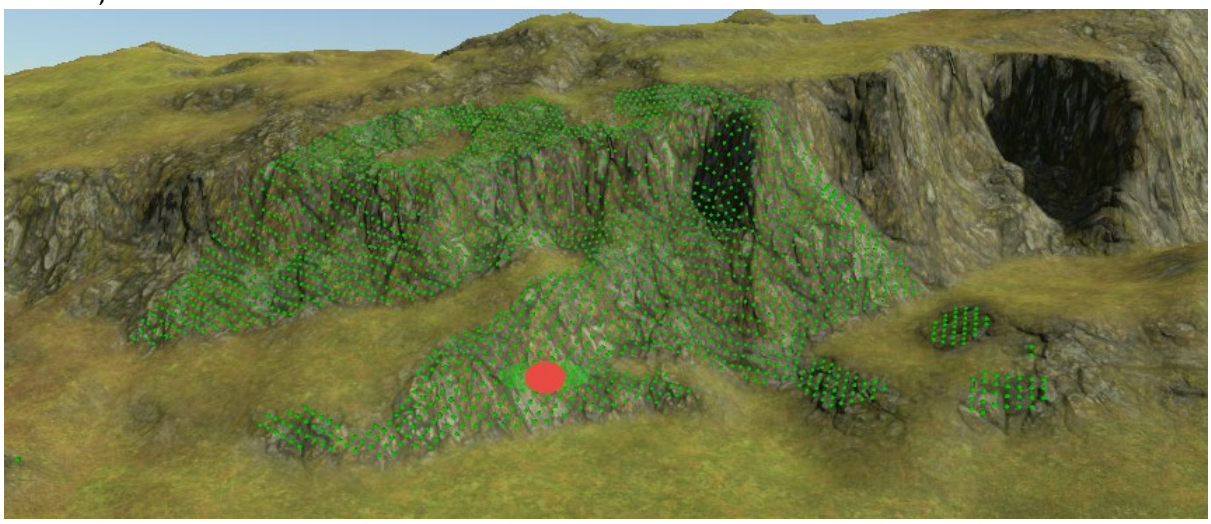
- ***None***: Do not use slope check.
- ***Range***: Spawns only within the slope range bisected by the slope of the point where the visualisation or spawn was initiated i.e., where you shift or ctrl left clicked on the terrain or mesh.
- ***Min Max***: The minimum and maximum slopes at which something will be spawned.
- ***Mixed***: The minimum and maximum sloped at which something will be spawned, then further restricted by slope range.
- ***Min Max Slope***: The minimum and maximum slopes at which something will be spawned.
- ***Slope Range***: Spawns only within in the slope range bisected by the slope of the point where the visualisation or spawn was initiated i.e. where you shift or ctrl left clicked on the terrain or mesh.

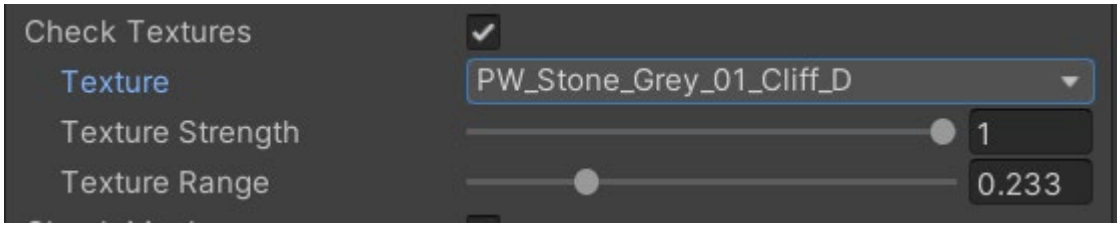


In this example, we have chosen a slope range of 26 degrees, and you can see approximately where on the terrain the shift click occurred. The visualiser shows the range of possible locations that it will consider for spawning that fall within the selected range.

Check Textures: Determines whether the terrain texture will be used as a range in which the spawner will spawn its Prototype.

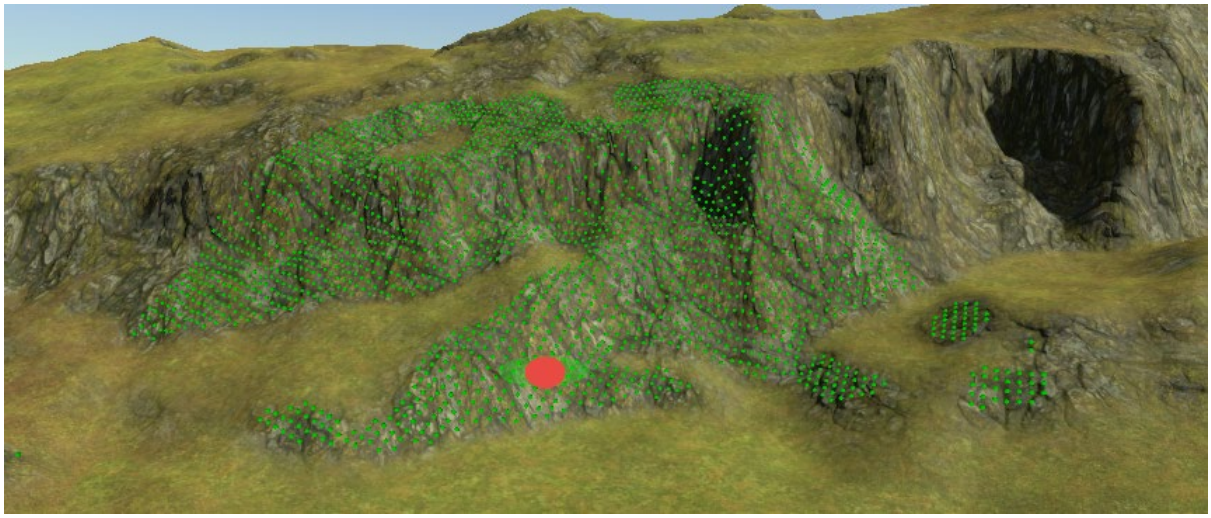
- *Texture*: The name of the texture that was clicked on.
- *Texture Strength*: The strength of the texture that was clicked on. Can be modified to select for other strengths.
- *Texture Range*: Spawns only within in the texture strength range bisected by the dominant texture of the point where the visualisation or spawn was initiated i.e., where you shift or ctrl left clicked on the terrain. This does not work for meshes.





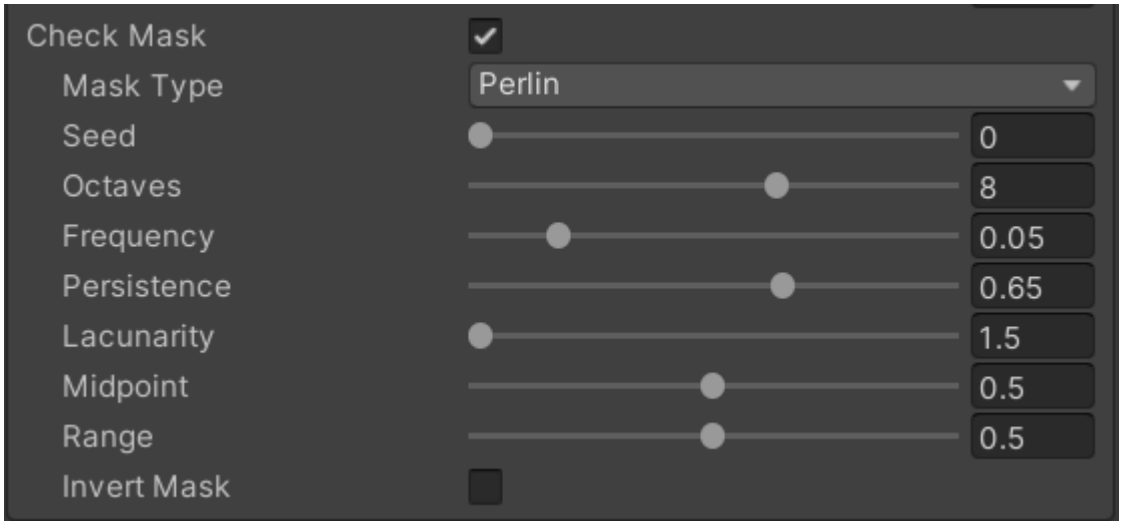
NOTE: You can change the texture strength and range to get some quite subtle effects.

- *Texture Range*: Spawns only within in the texture range bisected by the dominant texture of the point where the visualisation or spawn was initiated i.e. where you shift or ctrl left clicked on the terrain. This does not work for meshes.

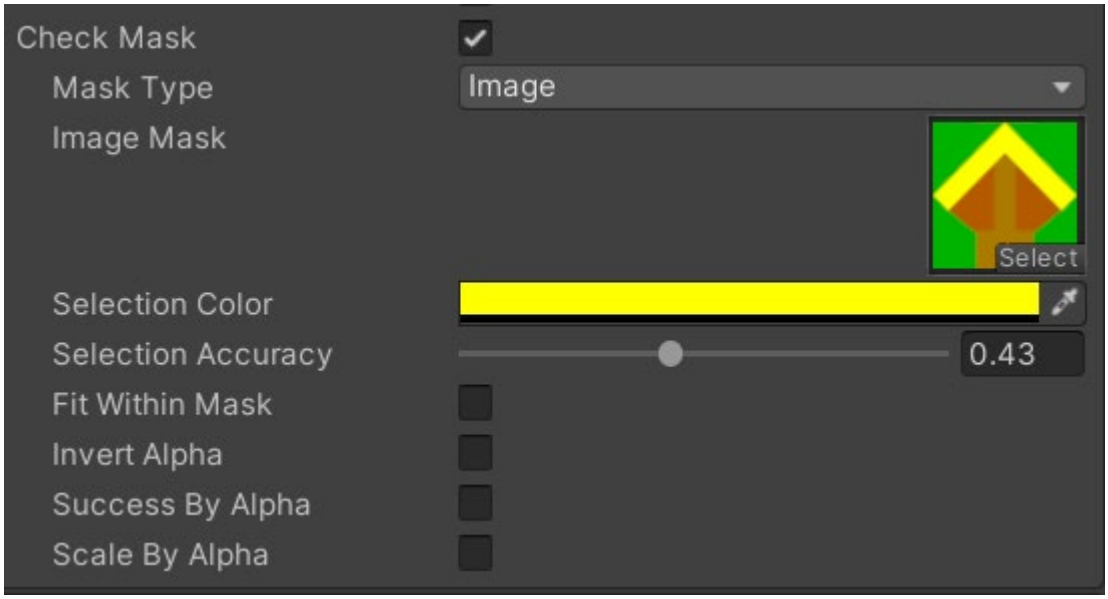


NOTE: You can change the texture strength and range that you test for to get some quite subtle effects.

Check Mask: Uses a mask to determine where the spawner will spawn its Prototype.



Noise based masks overlay noise on the spawner at a world coordinate level to determine where Prototypes can be spawned.



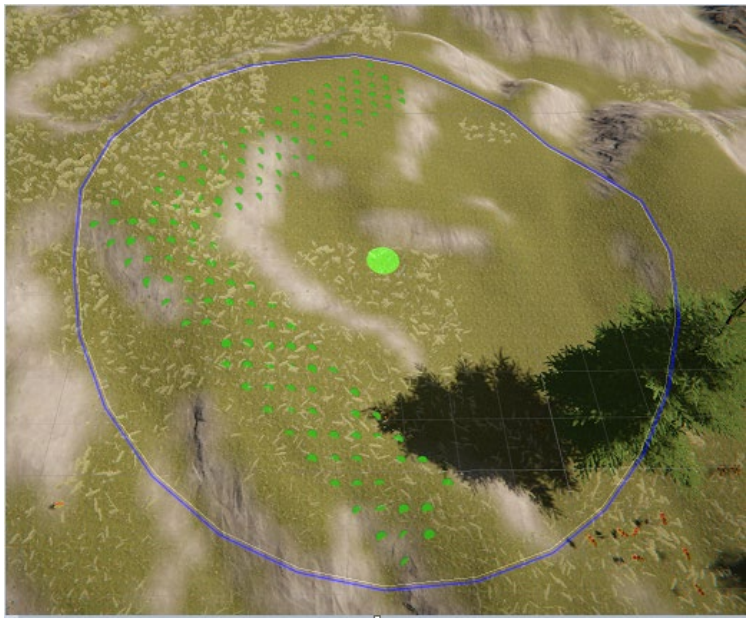


Image based masks use images to determine the location, scale and density of the Prototypes to be spawned.

Mask Type: The type of mask to use. The mask will be reflected in the visualiser.

- *Perlin, Billow, Ridged*: Noise masks applied at world scale.
- *Image*: Image mask applied at spawner scale.

Noise Masks:

- *Seed*: The unique seed.
- *Octaves*: The amount of detail in the noise - more octaves mean more detail and longer calculation time.
- *Frequency*: The frequency of the first octave. Smaller values create larger noise patterns.

- *Persistence*: The roughness of the noise. Controls how quickly amplitudes diminish for successive octaves.
- *Lacunarity*: The frequency multiplier between successive octaves.
- *Mid-Point*: The part of the noise curve that is treated as the midpoint of the noise function.
- *Range*: The range around the midpoint used to select the noise for spawning.

Image Masks:

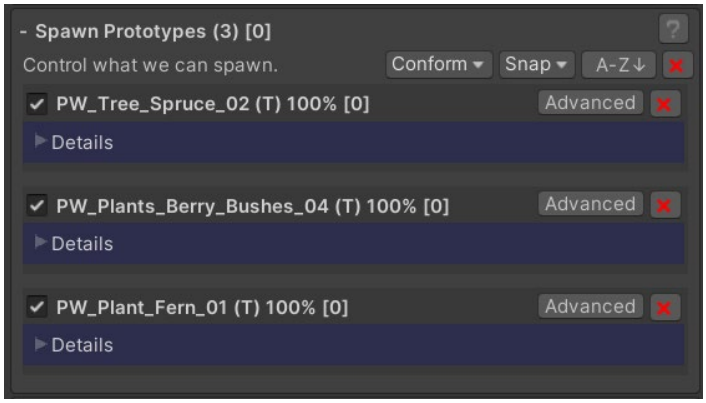
- *Image Mask*: The colour image to be used as a mask. The mask will be scaled to the range of the spawner. The primary colours of the image can have Prototypes assigned to them, and the alpha channel of the image if it exists can be used to control spawn size and density.

  You will select Prototypes against this image in the Spawn Prototypes section (see the relevant image masking section in there).

Spawn Prototypes Panel

This section defines the Protoypes and the settings that control how they will spawn into a scene.

A Prototype consists of one or more physical resources that can be spawned into a scene – such as Terrain Grass / details, Terrain Trees, and GameObjects / Prefabs.



Each Prototype will have one entry in the Spawn Prototypes panel.

They are identified as follows:

(Check) – If checked then the Prototype is active and can be spawned.

Name – Prototype name, prefabs will be parented under this.

(Tx/G/T/P) – Terrain Texture, Terrain Grass, Terrain Tree, (P)refab.

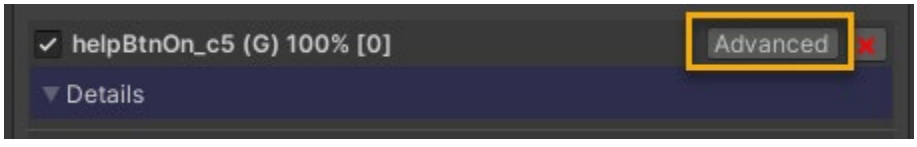
C – Conforms to terrain normal (only for prefabs)

% – Percent chance of being spawned if selected

(0) – The number of instances spawned.

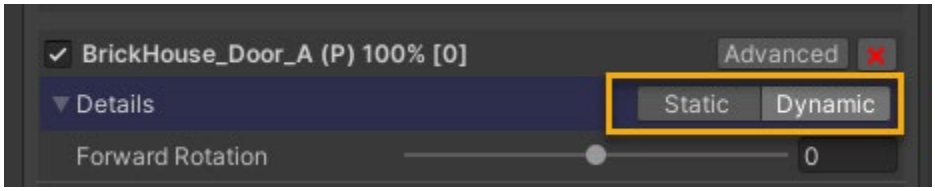
To see the detail of how the prototype has been configured open the detail tab by clicking on the triangle. The detail will vary depending on the type of prototype and whether it is being applied via an image mask.

Advanced View



Clicking on the Advanced View button will show a lot more detail about how the prototype has been configured.

Static / Dynamic View



Switches the view mode between static and dynamic.

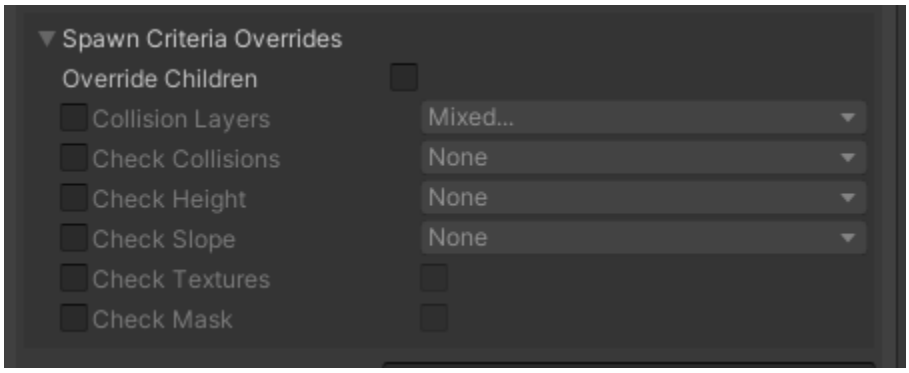
Static views are not designed to be edited and the settings are set automatically by the spawn ingestion process on sub objects in a Prefab / Game Object tree.

Dynamic views are designed to be editable and are generally at the root of a prototype tree.

You can switch a static view to dynamic if you want to customize its internal settings.

Spawn Criteria Overrides

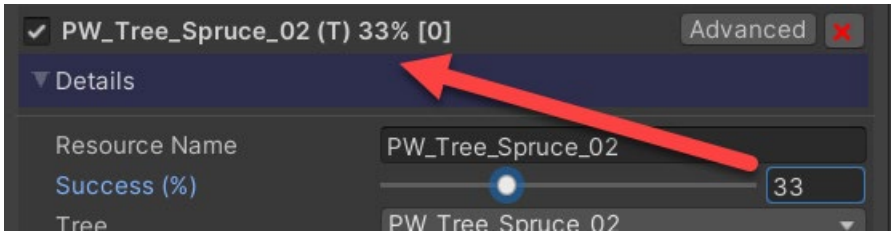
When in an advanced view, you can choose to override the spawn criteria for each individual resource. This can be useful on complex structures.



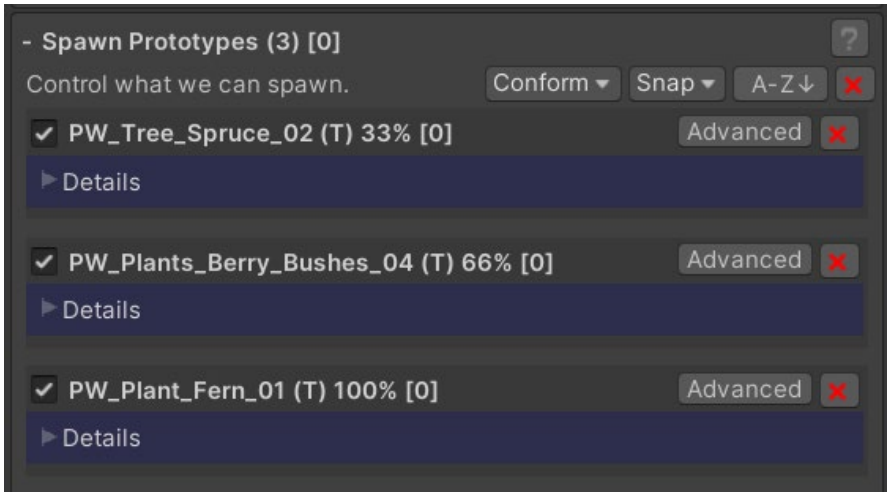
Success Rate (%)

When a spawn iteration occurs, GeNa will choose a location on the terrain, evaluate it for fitness to spawn, and then do a success check. If the prototype fails this check, then it will fail to spawn.

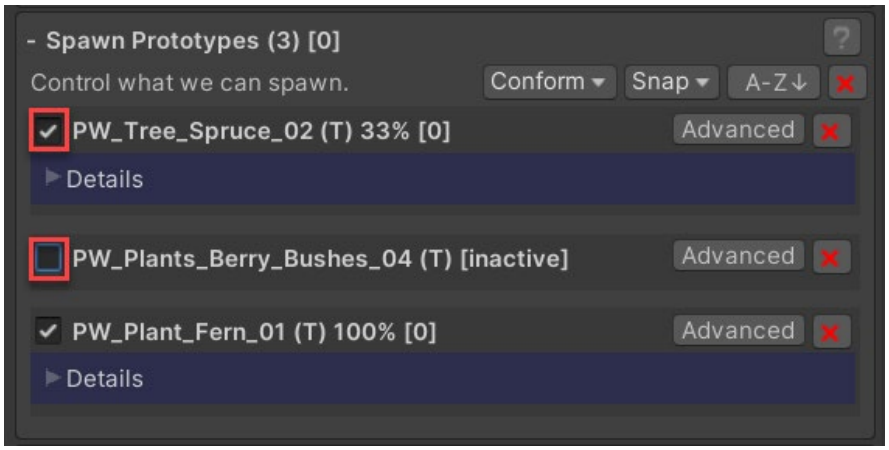
To change the ratios of how each resource spawns compared to another, change their success rates.



The success factor is shown as a percentage in the Prototype title.



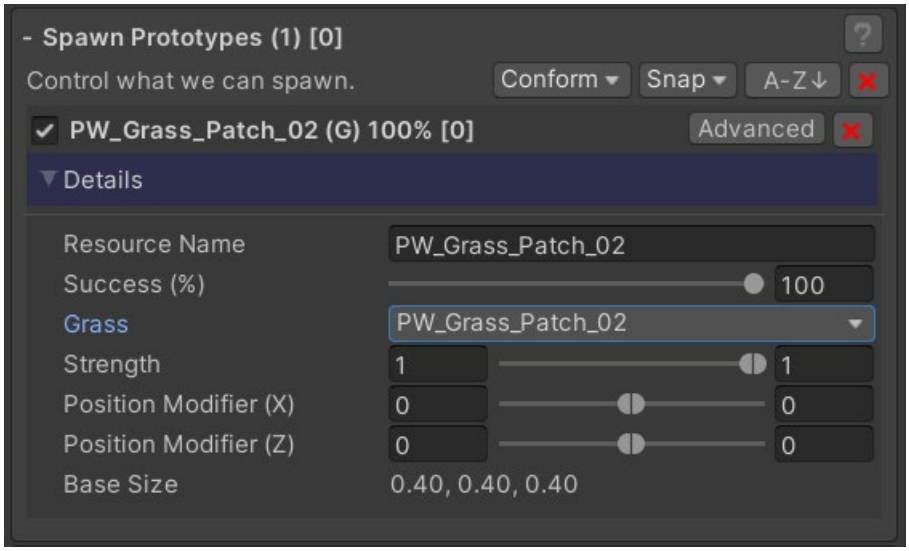
The relative success ratio influences the proportions of each object being spawned, as shown in the instance counts.



To enable or disable different resources to be considered for spawning just click the checkbox next to the resource. Inactive resources will be excluded from the spawn.

Grass Prototypes

Grass Prototypes describe terrain details and how they are applied by the spawner.



Resource Name: Informational name.

Success: Percentage change of a successful spawn. Zero means no chance, one means no problems.

Grass: A drop down that interrogates your terrain and allows you to select a grass. You must have previously added the grass to the terrain to select it here. See: https://docs.unity3d.com/Manual/terrain-Grass.html.

Strength: The minimum and maximum strength of the terrain grass to be spawned. This equates to target strength on the grass painting in your terrain settings.

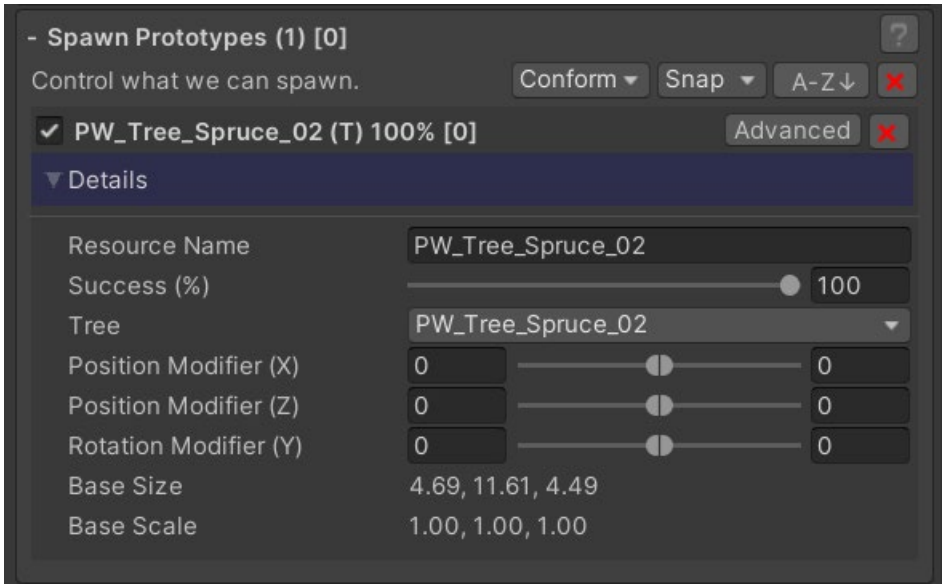
Position Modifier X: Offset from spawn location in x direction to spawn grass.

Position Modifier Z: Offset from spawn location in z direction to spawn grass.

Base Size: The basic size of a grass instance. An informational value that is pulled from the way it has been applied to the terrain.

Tree Prototypes

Tree Prototypes describe terrain trees, and how they are applied by the spawner. Trees can also be spawned as prefabs / game objects. See the prefab section for instructions on how to do this.



Resource Name: Informational name.

Success: Percentage change of a successful spawn. Zero means no chance, one means no problems.

Tree: A drop down that interrogates your terrain and allows you to select a tree. You must have previously added the tree to the terrain to select it here. See: https://docs.unity3d.com/Manual/terrain-Trees.html.

Min & Max Modifier (X, Y, Z): When spawning, a random offset between the minimum and maximum value will be added to the rotation selected for the tree in the Placement Criteria. In some scenarios you can want the tree to have a different offset than the one being applied to the spawn, particularly when using draggable rotation with complex POI spawns.

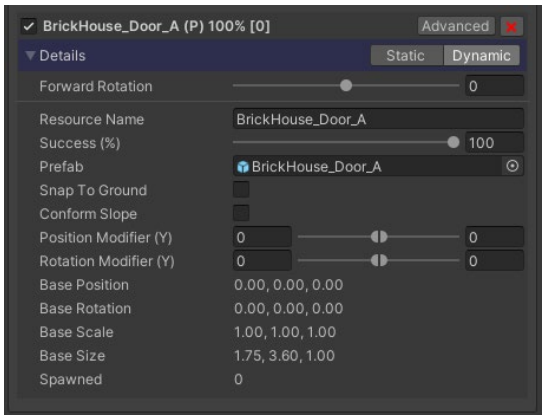
Base Size: The basic size of a tree instance. An informational value determined by instantiating an instance of the tree and calculating its renderer and collider bounds.

Base Scale: The scale of the original tree instance. An informational value picked up from the tree prefab.

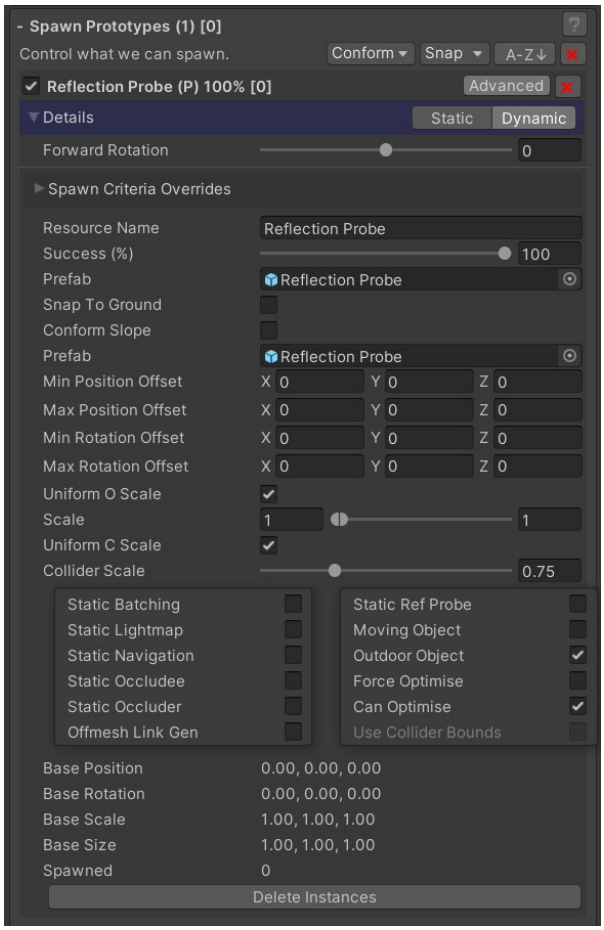
Prefab Prototypes

Prefab Prototypes describe prefabs or collections of prefabs (structures, villages etc), and how they are applied by the spawner.

Simple (non Advanced) view:



Advanced View



Forward Rotation: An additional rotation that will be added to the rotation in the Placement Criteria so that the prototype will always face in the direction of the arrow when draggable rotation has been selected. Allows you to get very precise control over the direction an object will face when it spawns.

Resource Name: Informational name. Used as the name of this prefab as it is spawned into the scene.

Success: Percentage change of a successful spawn. Zero means no chance, one means no problems.

Prefab: The prefab that will be spawned. This can be any prefab. NOTE: If you want this prefab to generate or avoid collisions then it must have a collider.

Snap To Ground: Snap the Prefab position to the closest point on ground.

Conform Slope: Conform the prefab to the normal of the slope on which it will be spawned.

Min / Max Position Offset: The offset from the spawn position that will be applied to this prefab when it is spawned. Can be used as a range so that an object can be positioned differently from spawn to spawn.

Min / Max Rotation Offset: The offset from the spawn rotation specified in the Placement Criteria that will be applied to this prefab when it is spawned. Can be used as a range so that an object can be rotated differently from spawn to spawn.

Uniform O Scale: Apply a constant scale override to the scale of the object spawned or apply a ranged scale. The scale of the spawned object will be the result of the Placement Criteria scale multiplied by this scale.

Min / Max Scale: Apply a random range for the scale of the object to be spawned.

Uniform C Scale: Apply a constant scale override to the scale of the collider on the object spawned when using gravity or apply a per dimension scale.

Collider Scale: The collider scale to apply when spawning objects for gravity.

Static Batching / Lightmap / Navigation / Occludee / Occluder / Offmesh Link / Static Ref Probe: Static overrides that can be applied when spawning. Can only be applied in editor mode, not during runtime spawning. See here for more information: https://docs.unity3d.com/Manual/StaticObjects.html.

Outdoor Object: This is an outdoor object. Blend probes will also use the skybox when spawn optimization is chosen. Indoor objects will not use the skybox.

Force Optimise: The object will use the optimization process if optimize bake is chosen in advanced settings.

Can Optimise: The object will be considered for optimization if optimize bake is chosen in advances settings.

Use Collider Bounds: Legacy. Not used.

Base Position: The base position of a prefab instance. This is typically the original positioning of the root of the prefab.

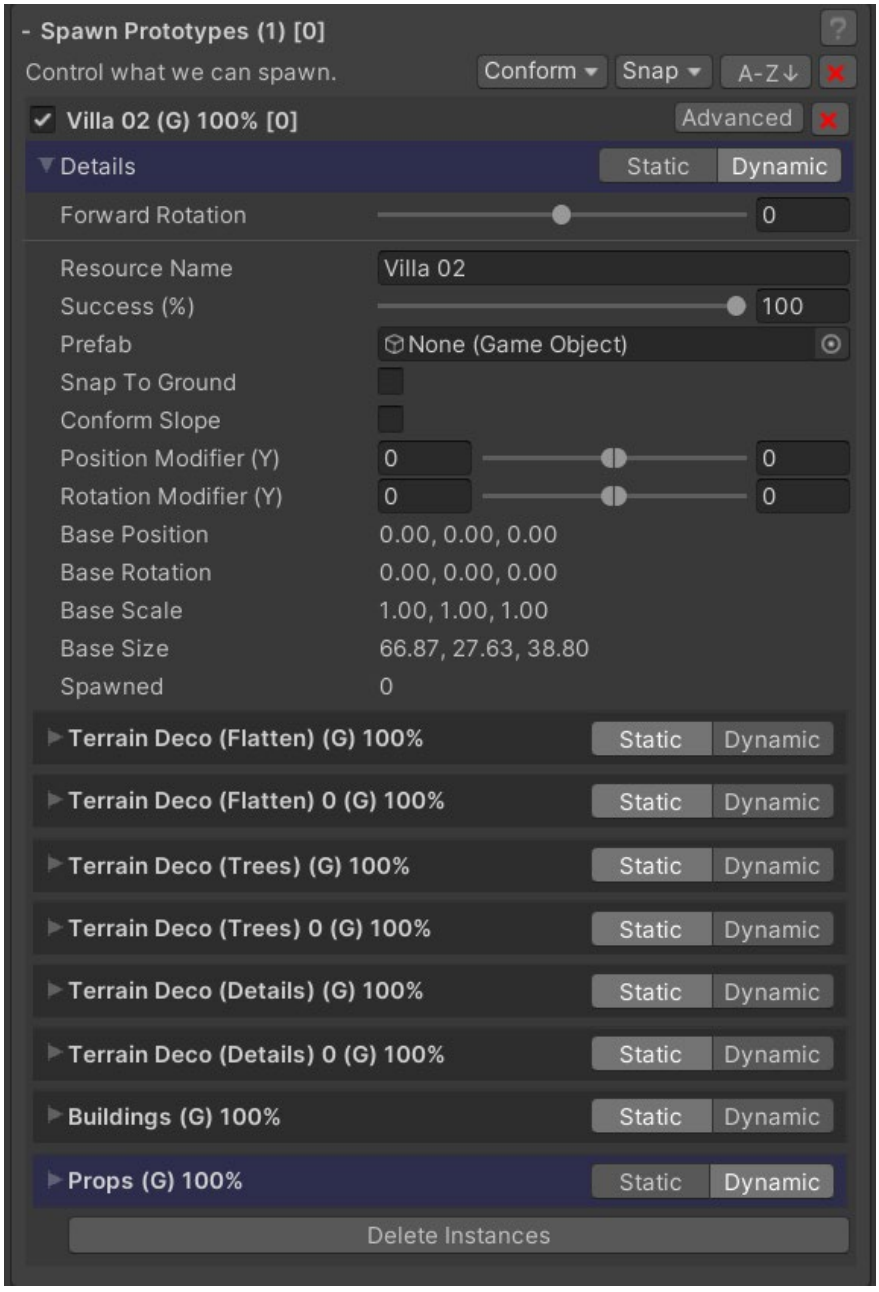
Base Size: The basic size of a prefab instance. An informational value determined by instantiating an instance of the prefab and calculating its renderer and collider bounds.

Base Scale: The original scale values for the prefab instance. An informational value derived from the original prefab.

Prefab Prototype Collections (POI's)

When prefabs are added as collections rather than singletons they will be spawned as collections and will maintain their layout relative to each other when they are placed into the environment.

This is a cool way to get things like farms and graveyards. You can design them once, and then place them as many times as you like.

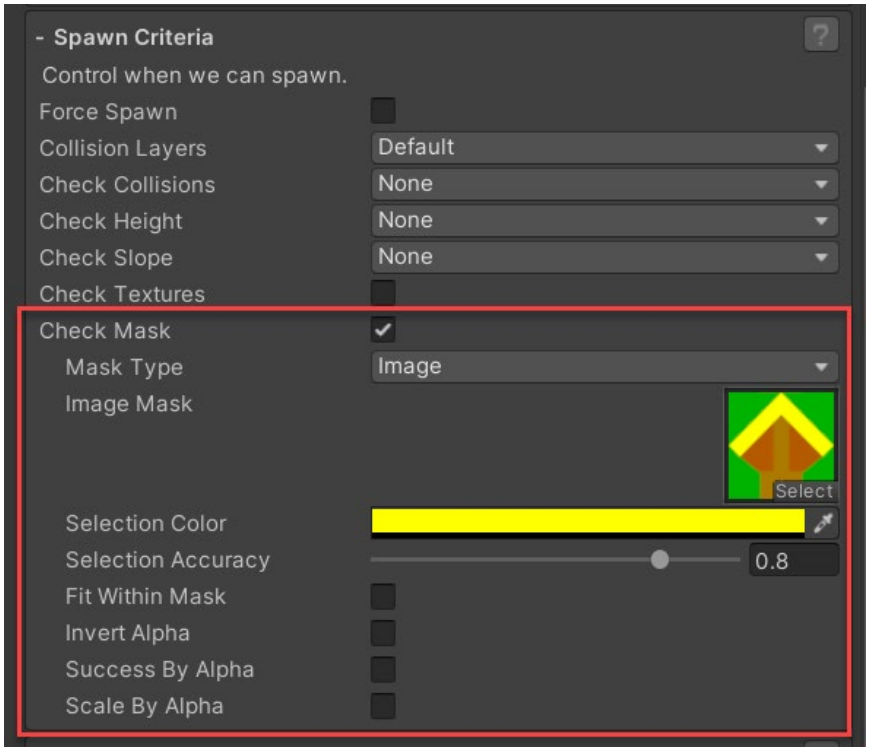


The Forward Rotation setting allows you to pre-rotate the entire POI so that the forward arrow will always generate the right "forward" rotation when draggable rotation has been set in Advanced Settings. This allows for very precise positioning of your POI.

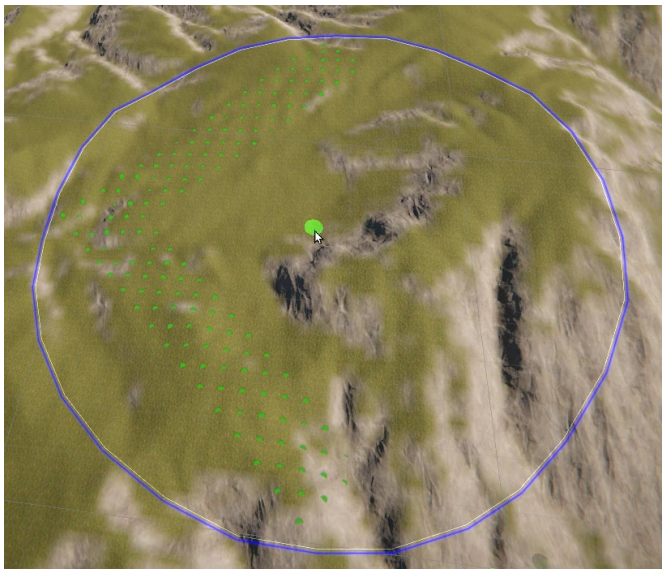
In the image above you can see that this prototype has many resources. You can change the settings of each resource individually – refer to the previous Prefab Prototypes sections for more detail.

Image Masking Settings

When you select choose to use an Image Mask, a new section called Mask Settings will show on your Prototype and allow you to key your Prototype into the colour on the mask that you want to have it spawned on, and then set the rest of the spawn related settings.



A nice way to do this is to select the colour dropper and then click on the preview image on the colour you want to use.



As you key each new Prototype into the mask, it will appear in the visualiser (you may need to shift left click again to see it). You can activate and inactivate individual resources and see the impact of this in your visualiser.

Selection Colour: The mask colour this Prototype will spawn on. Multiple Prototypes can be keyed to the same colour.

Selection Accuracy: The accuracy of the colour selection. In general you should never change this.

Fit Within Mask: When doing bounds based spawning this will ensure that the entire volume of the object being spawned will be contained within the mask. Good for buildings for example, but not so good for trees. It will depend on your use case.

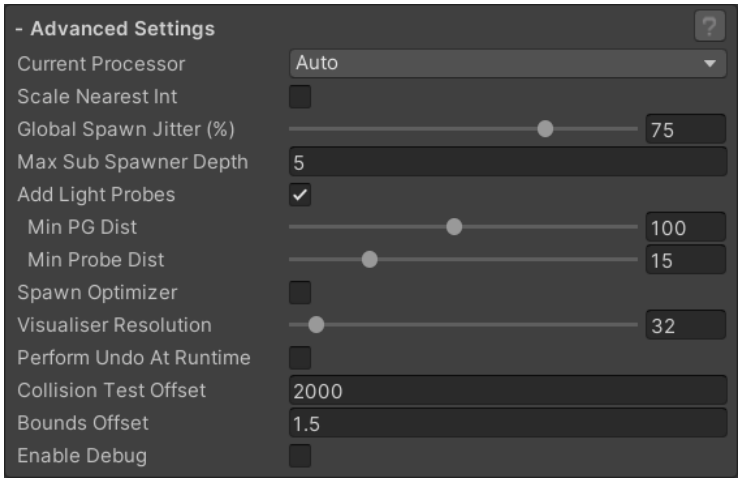
Invert Alpha: Invert the alpha channel. Can be useful when making Prototypes spawn in mutually exclusive places off one mask.

Success By Alpha: The prototype success factor will be influenced by the alpha channel. Works in the same way as the Prototype success control.

Scale By Alpha: The prototype scale will be influenced by the alpha channel. Works in the same way as the Prototype scale control.

Min / Max Scale: The range that the alpha value will be scaled do. An alpha value of zero will be the minimum value, and an alpha value of one will be the maximum value.

Advanced Settings Panel



This section defined the advanced settings for the spawner. Less commonly used spawner settings are placed here.

Current Processor:

Scale Nearest Int: Will force your objects to spawn to a scale that is based on the nearest whole number.

Global Spawn Jitter (%): The amount of jitter that will be introduced to where global spawns are initiated.

Max Sub Spawner Depth: The default depth of recursive Sub Spawner Spawn Calls.

Add Light Probes: This will cause light probe stacks to be added to the scene when a prefab is successfully spawned. A light prob stack is added ½ a meter above the spawn point, then ½ a meter above the top of the prefab, and then another 5 meters above that to ensure good light coverage.

GeNa will ensure that light probe groups are placed at least "Min PG Dist" meters away from each other, and that individual light probe stacks to be placed at least "Min Probe Dist" meters away from each other.

Spawn Optimizer: This will cause the spawner to optimize the way prefabs are placed during spawning. The optimizer will optimize all resources that fit its criteria. To be considered for optimization the resource must have "Can Optimize" set and be smaller than the "Smaller Than (m)" size (after scaling). Alternatively, if "Force Optimize" is set then it will always be optimised. See the section on Optimisation for more information.

Visualiser Resolution: The dimension of the points that will be checked for every time the visualiser evaluates the terrain. A value of 50 means that 50 x 50 = 2500 locations will be checked every time you update the visualiser – the bigger the value the more time this takes. If the time to calculate this exceeds 200 milliseconds GeNa will automatically reduce

this. This ensures that the Unity editor remains responsive regardless of the power of the computer it is running on.

Perform Undo at Runtime: Enables the ability to perform undo at runtime.

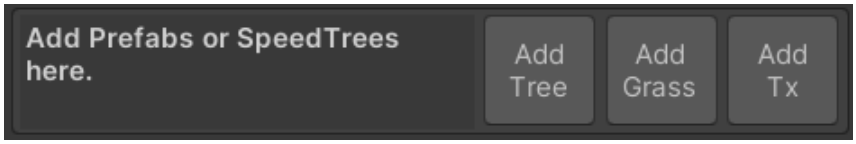
Collision Test Offset: Collisions are based on where you click on the screen. This offset is added to the location where the hit occurred, and a new collision check is then initiated from the new height. This allows you to detect if your object is under an overhand. Sometimes however, your object is in an enclosed space, so making this much smaller will stop your collision check from hitting overhangs.

Bounds Offset: Offset from spawn range to detect the nearest collision objects.

Enable Debug: Enables debugging for the Spawner.

Add Resources Panel

This section allows new resources to be added to the spawner.



Add Grass: Click this to add a Terrain Grass Prototype. It will default to always select the first grass in the terrain. Go in and edit the prototype to select the right grass.

Add Tree: Click this to add a Terrain Tree Prototype. It will default to always select the first tree in the terrain. Go in and edit the prototype to select the right tree.

Add Prefabs or SpeedTrees here: Drag and drop Game Object Trees, Prefabs, or SpeedTrees here to add Prototypes.

There are multiple ways to ingest different types of structures in your Spawner.

Decorators

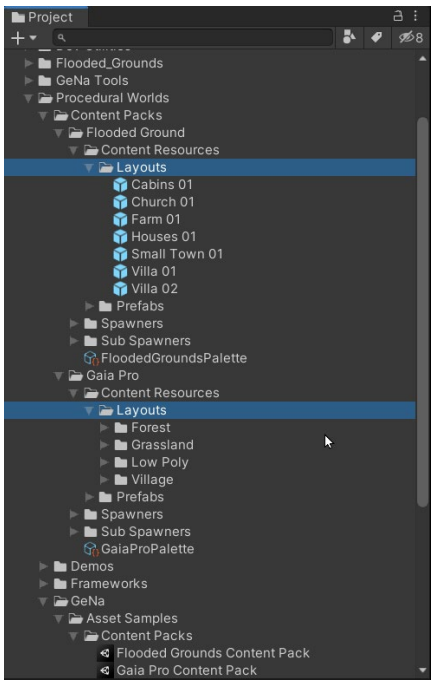
The Spawner system can be extended with the **Spawner Decorator** system.

Decorators are scripts you add to game objects or prefabs before ingesting them into a Spawner that influence the way the spawner operates.

For example, a Terrain Decorator can flatten the terrain under a building when it is spawned into the scene. The Decorator system is extensible so you can create and add your own customised spawn behaviour.

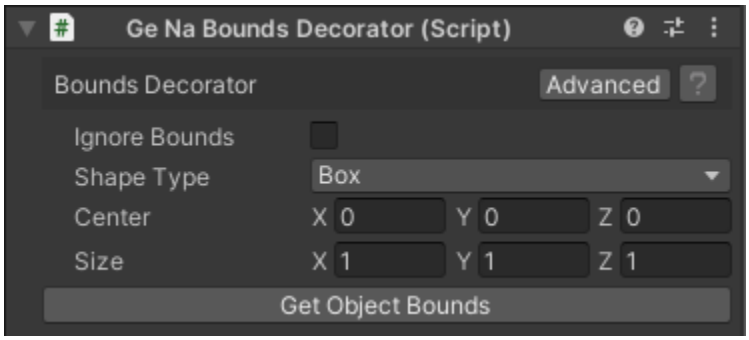
If you would like to see more examples of using Decorators, you'll find some examples of these under the following folders. Please see Content Pack Installation & Usage.

- **Procedural Worlds / Content Packs / Flooded Grounds / Content Resources / Layouts**
- **Procedural Worlds / Content Packs / Gaia Pro / Content Resources / Layouts**



Bounds Decorator

Allows an object to be ignored by the collision detection system, or alternatively, to be treated differently by the collision detection system.



Ignore Bounds: Should this object be ignored by bounds detection?

Shape Type: Choose between Sphere, Box, Capsule, Cylinder

Radius (Sphere, Capsule, Cylinder): Radius of the shape.

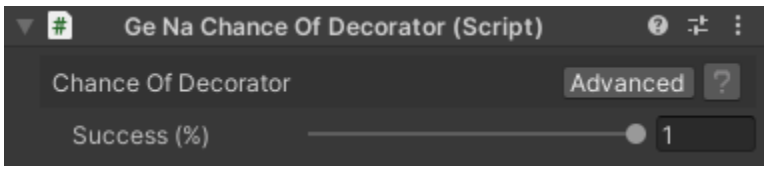
Height (Capsule, Cylinder): Height of the shape.

Center: Center of bounds.

Size: Size of bounds.

Chance Of Decorator

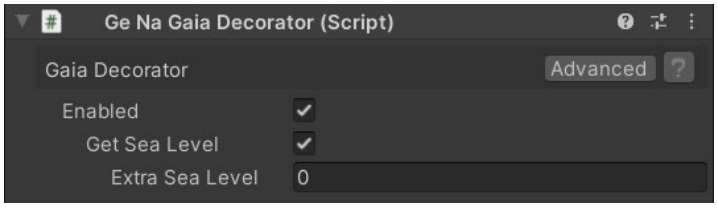
Influences the success rate of a game object or prefab.



Success (%): Success rate percentage of object spawning.

Gaia Decorator

Allows Gaia to be integrated if it is in the scene.



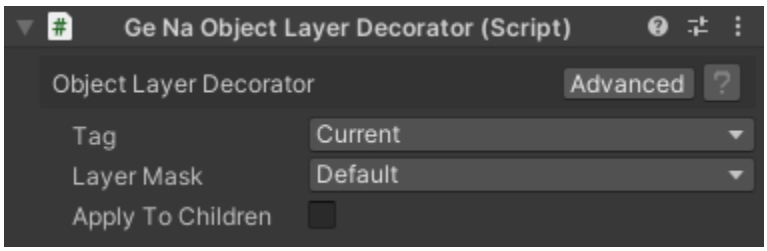
Enabled: Legacy. Always set to on.

Get Sea Level: If enabled, the spawner will always get the latest Sea Level from Gaia when spawning.

Extra Sea Level: Allows you to add or remove more height from the base Sea Level from Gaia.

Object Layer Decorator

Allows you to set the tag / layer mask of objects spawned into the scene.



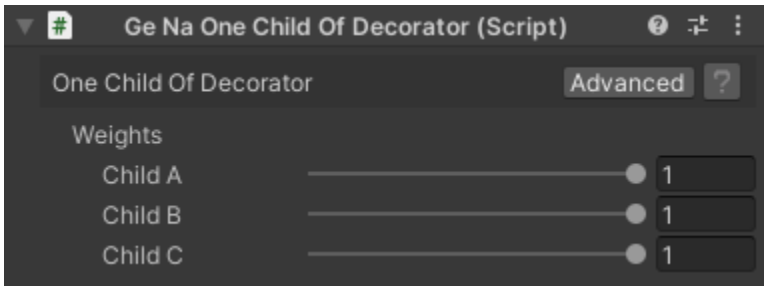
Tag: Sets the Tag of the GameObject.

Layer Mask: Sets the Layer of the GameObject.

Apply To Children: If enabled, this will apply the Tag and Layer to all the Children GameObjects.

One Child Of Decorator

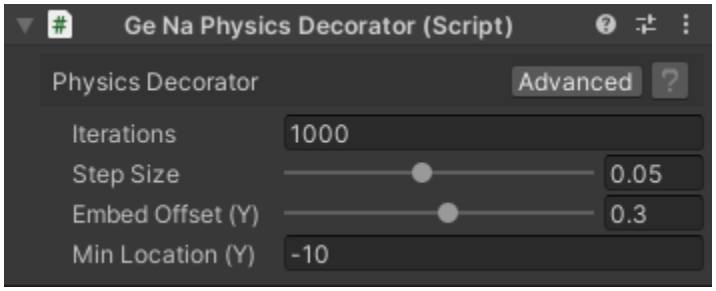
This decorator will cause GeNa to randomly select and spawn only one of the child objects or prefabs based on the ratio provided and will ignore the rest.



Weights: The weighting percentage of each child transform to select.

Physics Decorator

Allows you to cause this object to be spawned with physics enabled.



Iterations: Frame iterations to perform Physics Simulation

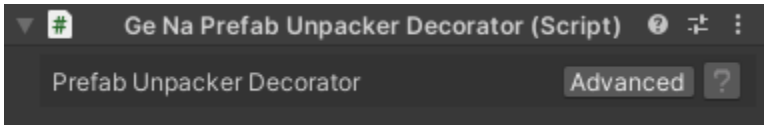
Step Size: Time delay between Physics Iteration.

Embed Amount (Y): Amount to embed object after physics simulation.

Min Location (Y): The minimum Y location a physics object can travel before exiting the simulation.

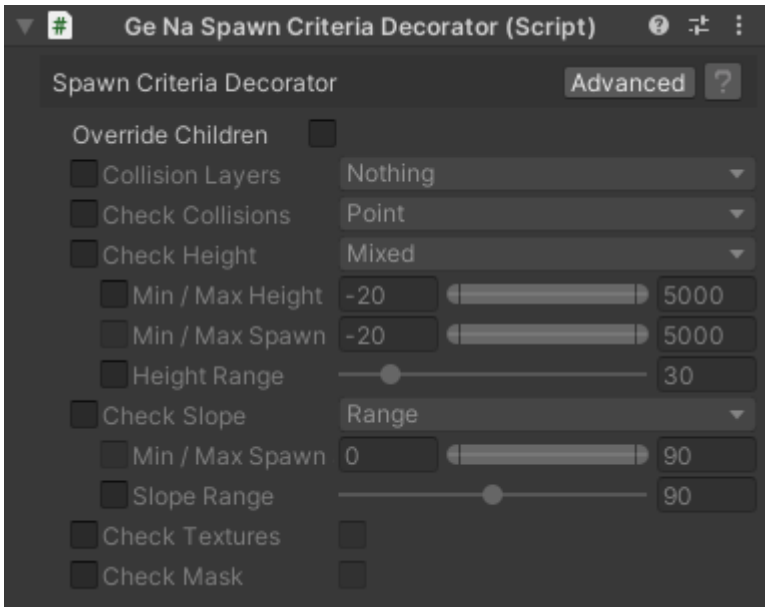
Prefab Unpacker Decorator

Calls Unity's prefab utility to unpack prefabs in the object tree so that GeNa can ingest their data correctly. If you have prefab unpack decorators within the children of the root object, then the root object will require a prefab unpack decorator.



Spawn Criteria Decorator

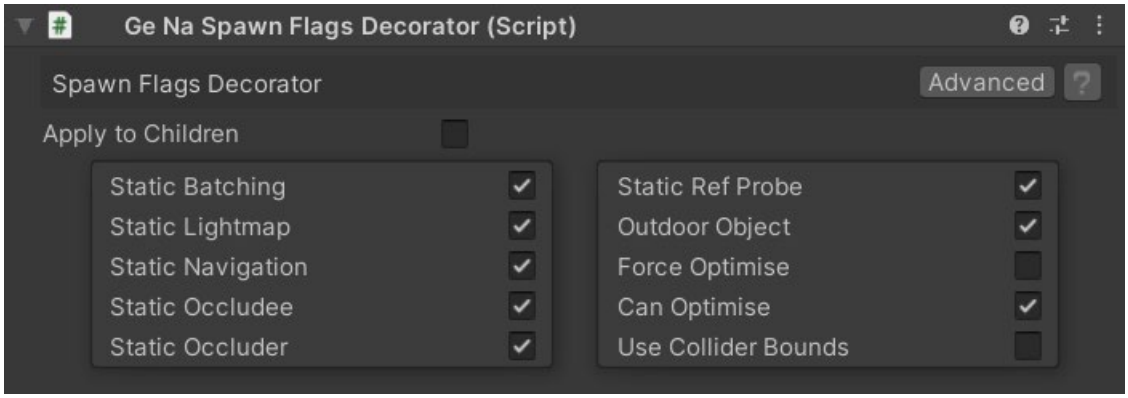
Overrides the spawn criteria on the object.



Please See Spawn Criteria Panel for more information.

Spawn Flags Decorator

Overrides spawn flags on an object.



Apply to Children: Apply all the configured settings to the children (recursive).

Static Batching: Spawn the prefab with or without static batching flag.

Static Lightmap: Spawn the prefab with or without static lightmap flag.

Static Navigation: Spawn the prefab with or without static navigation flag.

Static Occludee: Spawn the prefab with or without static occludee flag.

Static Occluder: Spawn the prefab with or without static occluder flag.

Static Ref Probe: Spawn the prefab with or without static reflection probe flag.

Outdoor Object: This is an outdoor object, blend probes will also use the Skybox with spawning, otherwise will exclude Skybox.

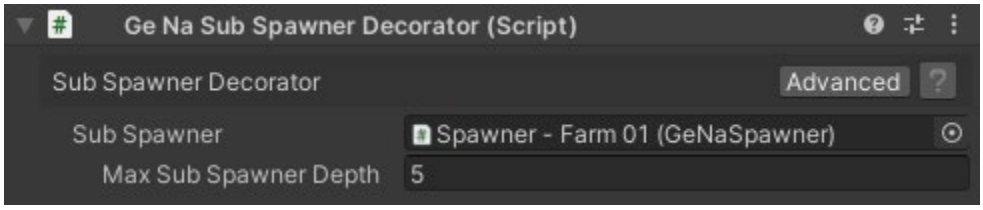
Force Optimise: The prefab will be forced through the optimisation process if Optimize Bake is chosen in Advanced Settings.

Can Optimise: The prefab will be considered for optimisation if Optimize Bake is chosen in Advanced Settings.

Use Collider Bounds: If enabled, the collider will be used for bounds checking.

Sub Spawner Decorator

Calls spawns another spawner at this location.

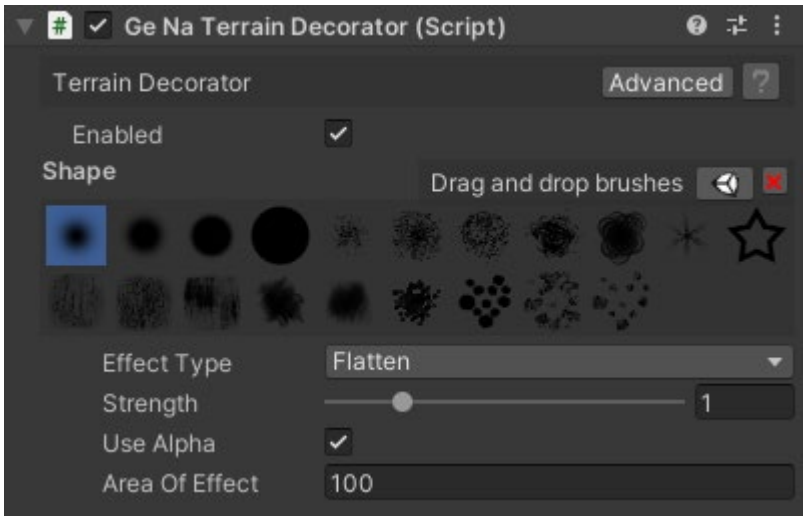


Sub Spawner: Attached Sub Spawner Object.

Max Sub Spawner Depth: How deep the Spawn Operation will run for Sub Spawner.

Terrain Decorator

Calls terrain operations at the location.



Enabled: If enabled, the Terrain Decorator will perform operations upon spawning GameObject.

Shape Textures: This area contains all Textures to be used for textures.

Effect Type:

- **Raise:** Raises the Terrain.
- **Lower:** Lowers the Terrain.
- **Flatten:** Flattens the Terrain.
- **Clear Trees:** Removes Terrain Trees.
- **Clear Details:** Removes Terrain Details.

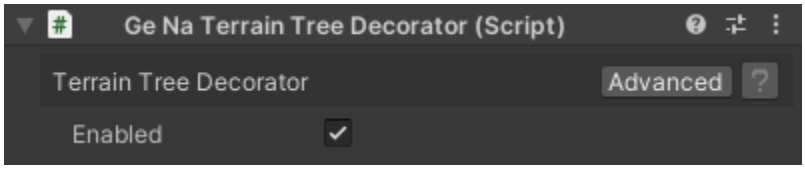
Strength: Strength of the Texture to perform operation.

Use Alpha: If enabled, the Alpha channel will be used.

Area Of Effect: Diameter of Texture to World.

Terrain Tree Decorator

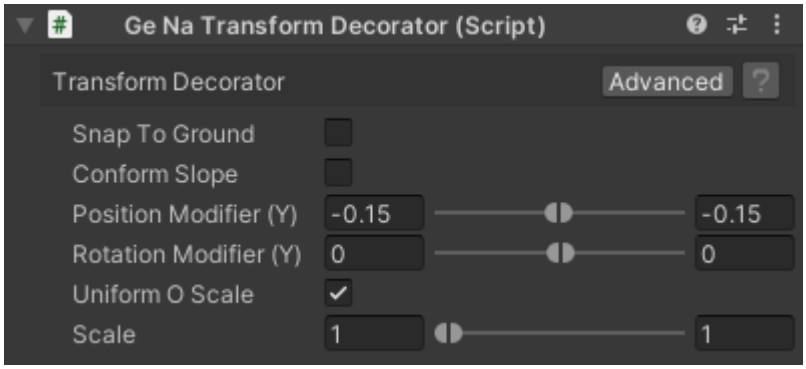
Adds the prefab that it is attached to, to the terrain as a terrain tree.



Enabled: If enabled, the GameObject will be added to the Terrain and spawned as a Terrain Tree.

Transform Decorator

Changes the position, location and scale of this object when it is spawned into the scene.



Snap To Ground: If enabled, the GameObject will snap to ground.

Conform Slope: If enabled, the GameObject will rotate to ground normal.

Position Modifier (Y): Position offset in the Y axis (see advanced for more).

Rotation Modifier (Y): Rotation offset in the Y axis (see advanced for more).

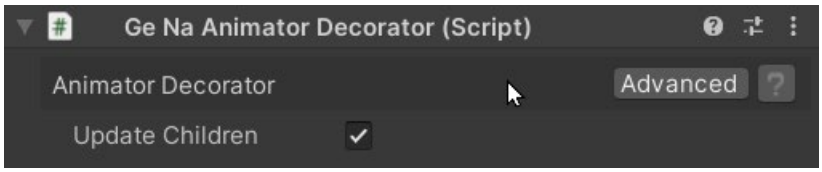
Uniform O Scale: If enabled, scale will be uniformed (X, Y, Z).

Scale: Scale Factor to apply to the GameObject.

Animator Decorator

Adds the ability to Animate your GameObject upon Spawning (works in both Edit & Play Mode).

Attach this to the **root** object of your Prefab to enable all animations.

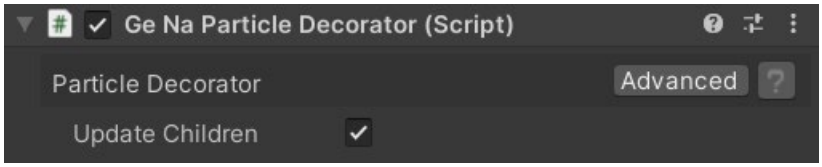


Update Children: If enabled, each of the Children will also perform Animations.

Particle Decorator

Adds the ability to Simulate Particles on your GameObject upon Spawning (works in both Edit & Play Mode).

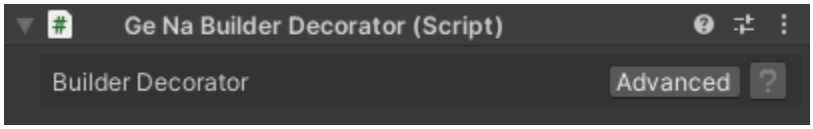
Attach this to the **root** object of your Prefab to enable all animations.



Snap To Ground: If enabled, each of the Children will also perform Particle Simulations.

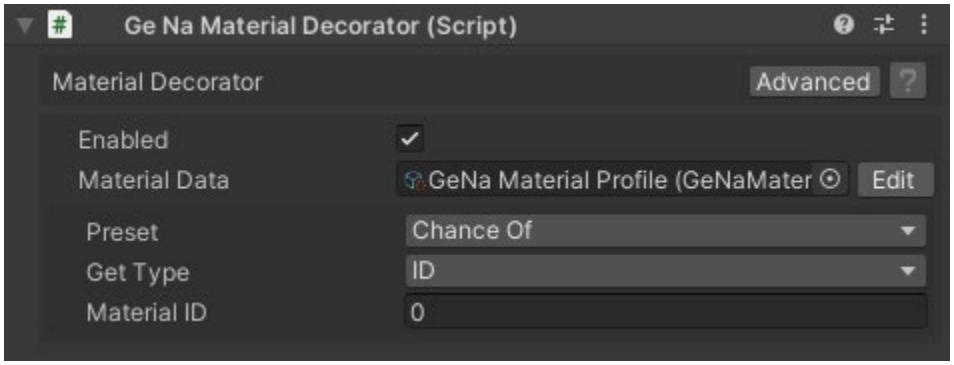
Builder Decorator

Draws Visualisation for assisting in designing new Prototype layouts in the Scene view.



Material Decorator

Allows you to swap out other materials based on different presets.



Material Data: The material data holds all the material information that will be used after the gameobject has been spawned. Within this you can setup material presets and add materials to the preset with a chance off selecting one of those material to use.

Edit: Selects and pings the material data object in the project window view.

Preset: Allows you to select a preset from the material data. If 'Chance Of' is selected it will pick one of the presets at random based off the chance off and then select one of the materials from that preset.

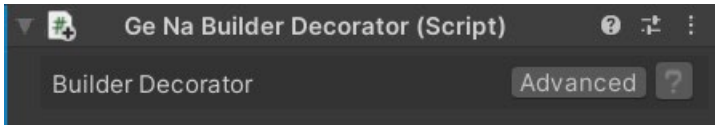
Get Type: Selects the get material ID.

> **Name**: Allows you to create a list of names that the material might have like: Leaf, Leaves etc. The name check is a contains 'Material Name' so only part of the name is required.
>
> **ID**: Selects a ID with a int from the materials from the mesh renderer on the gameobjects within the child.

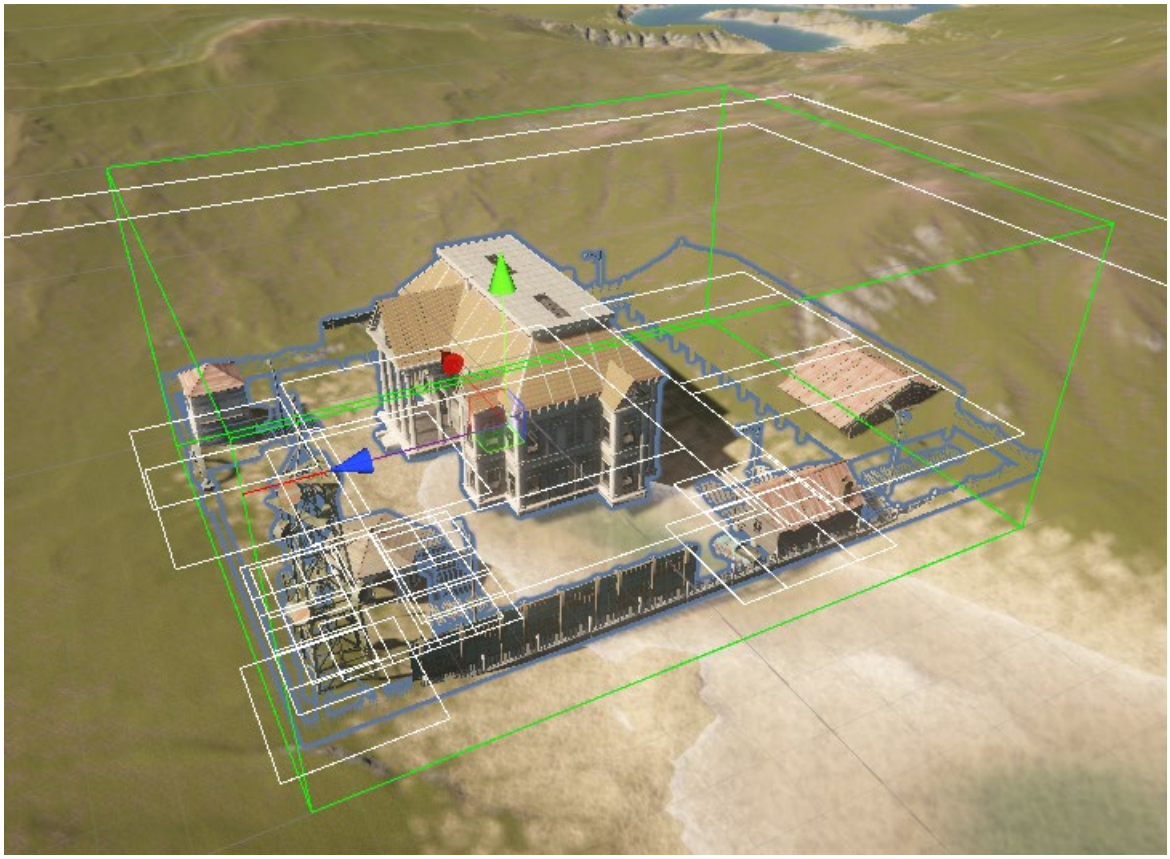
Material ID: Selects a ID with a int from the materials from the mesh renderer on the gameobjects within the child.

Usage

This section details some of the common usage scenarios for GeNa and makes some suggestions on how to get better results. NOTE: Ensure you attach this script to the **root** object of your Prototype.



This helps to visualise the bounds around separate parts of the prototype you're designing. Along with the bounds for the Flatteners, Sub Spawners and Collisions.

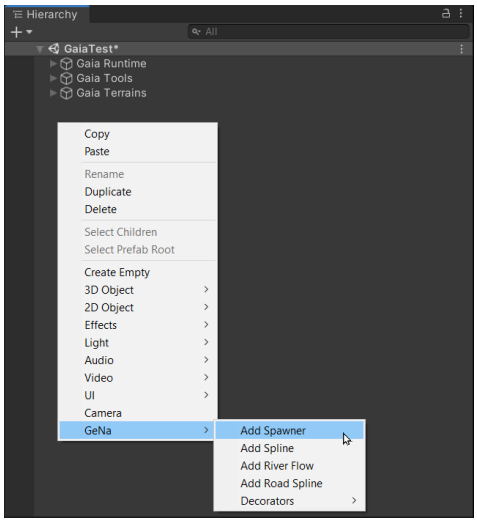


Create a GeNa Spawner

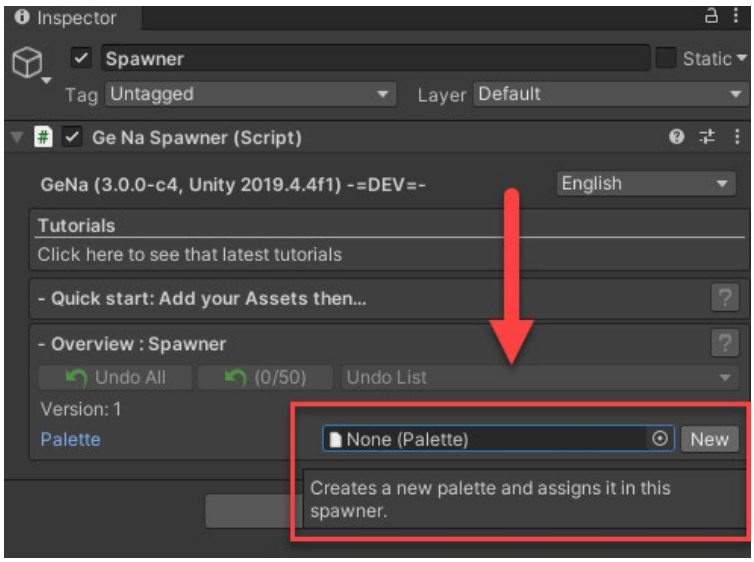
Step 1). **Create a new Spawner**

Right-click on an empty place in your hierarchy and select GeNa ->Add Spawner. Alternatively select Window -> Procedural Worlds -> GeNa Manager and press the Create Spawner button.

GeNa > Add Spawner



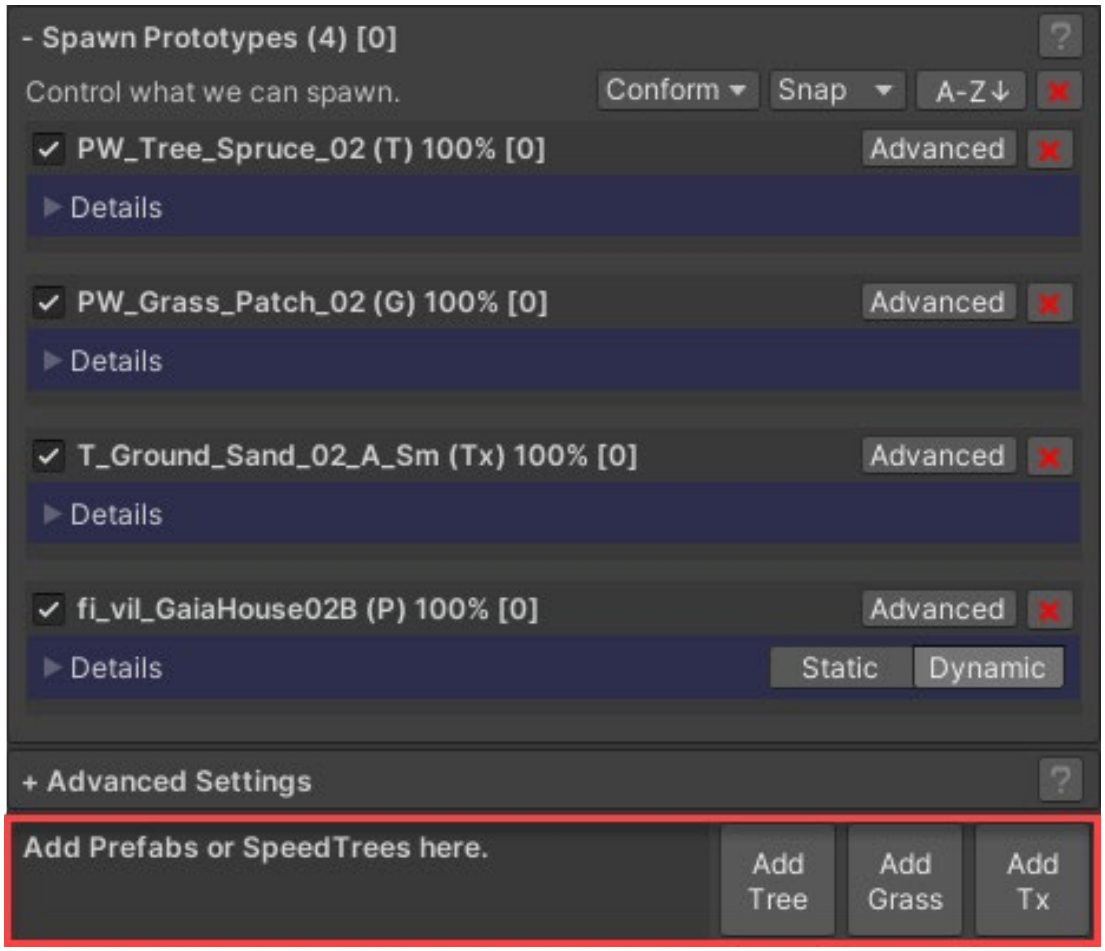
Step 2). **Select or Create a Palette**



Palettes maintain references to Assets for Serialization.

Step 3). **Add some Prototypes**

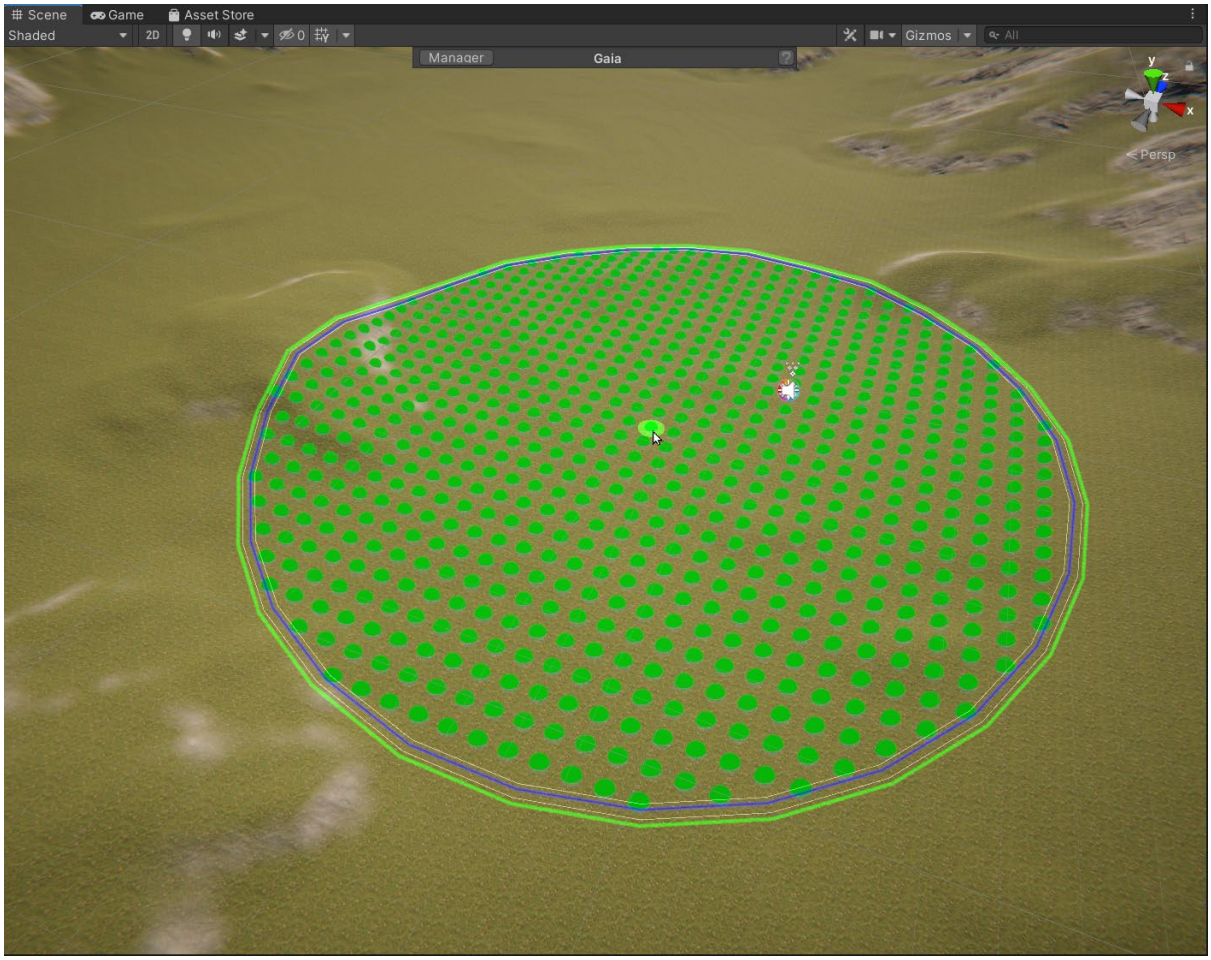
For GeNa to spawn things, you must first specify what type of object you wish to spawn. This is referred to as a Spawn 'Prototype'.



In this example, we have added the 4 different types of Prototypes. The terrain prototypes were added by clicking on Add Tree, Grass, Tx buttons and selecting the relevant asset in the prototype settings, and the prefab was added by dragging and dropping it onto the 'Add Prefabs or SpeedTrees' box.

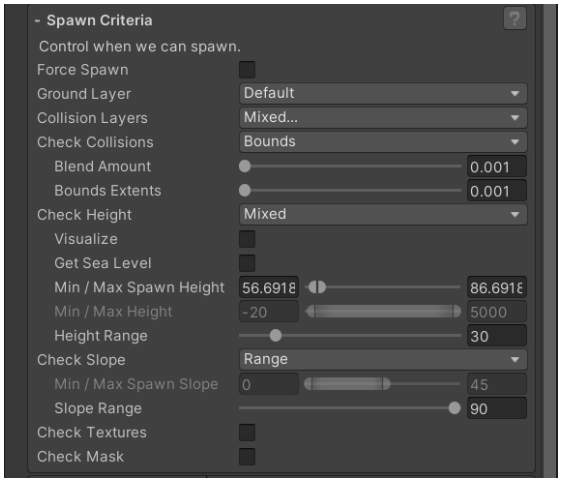
1. Terrain Tree - **PW_Tree_Spruce_02 (T)**
2. Terrain Grass – **PW_Grass_Patch_02 (G)**
3. Terrain Texture – **T_Ground_Sand_02_A_Sm (Tx)**
4. Prefab – **fi_vil_GaiaHouse02B (P)**

Step 4). To Spawn, we need to set our ground object by Shift + Left-Clicking on an object in our Environment. In this case, we will be spawning it on a Terrain.



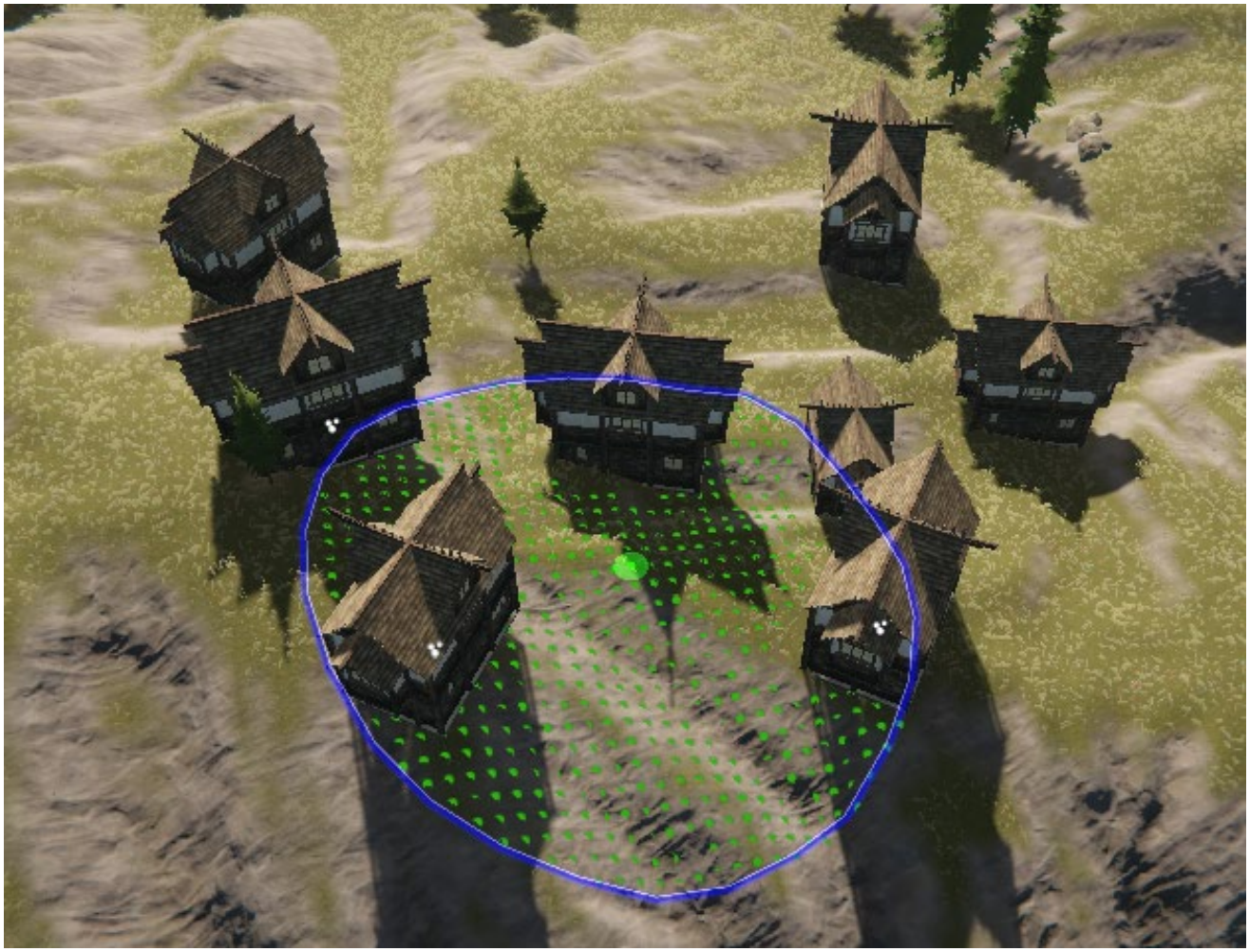
When you press and hold the **Shift** key, you will see the Spawner Visualizer. This tells us where we are allowed to spawn the Prototype based on our Spawn Criteria.

You can modify your Spawn Criteria in the Spawner to adjust this. Check the Spawn Criteria settings section of this for more information on this.



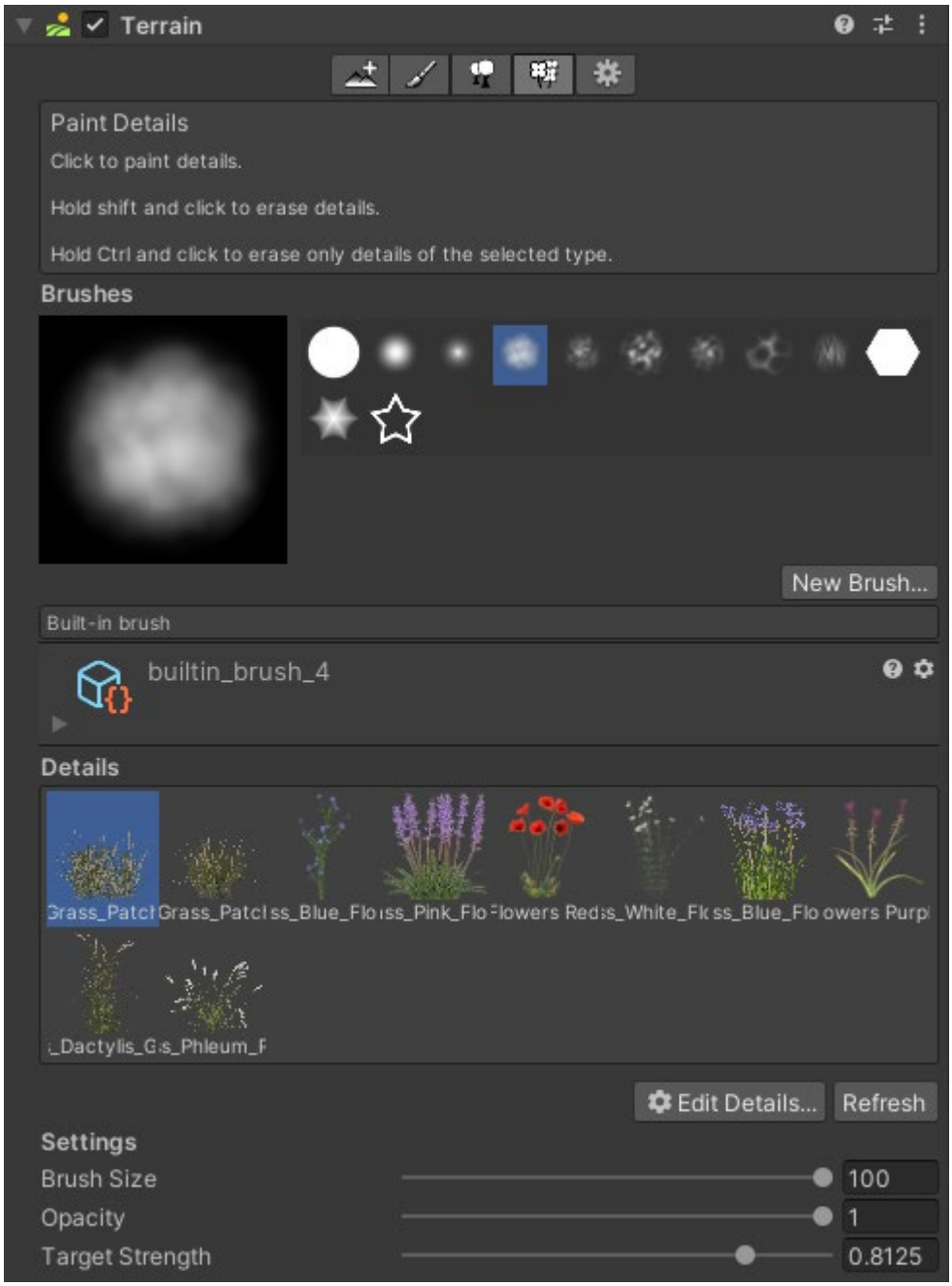
Step 5). **Spawn!**

Make sure your Spawner is selected in the hierarchy and press **Ctrl + Left click** to Spawn!

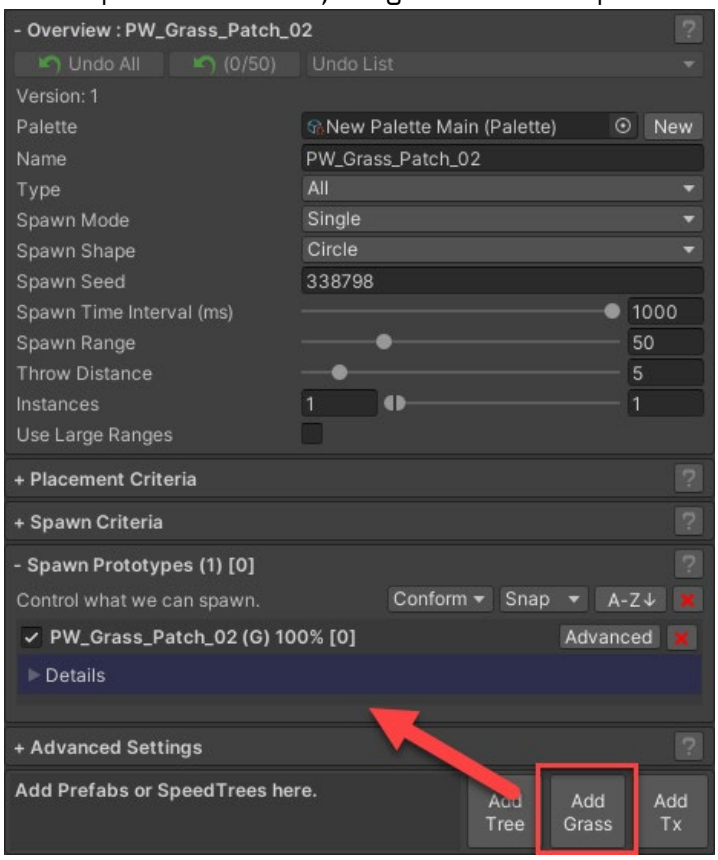


Spawning Terrain Details (Grass)

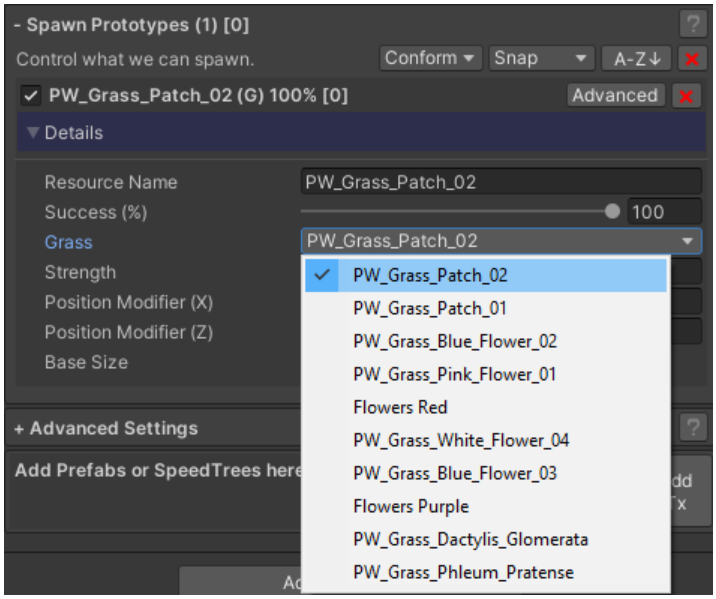
1. Add grass to your terrain.



2. Add a spawner and add your grasses to the spawner.



3. Go through each prototype and make sure the right grass is selected.



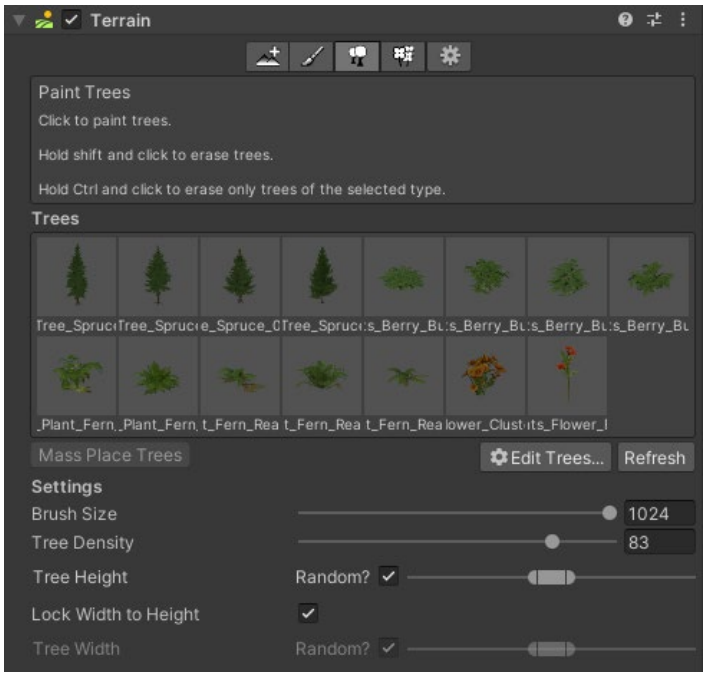
4. Spawn as usual.

TIP: The noise generator generates noise in world space, and this means that multiple spawners can use it and expect it will generate the same mask. Try setting up two spawners with the same noise settings, and one, select Invert Mask. This will invert the generated pattern and you can use this to spawn one grass in one section, and another

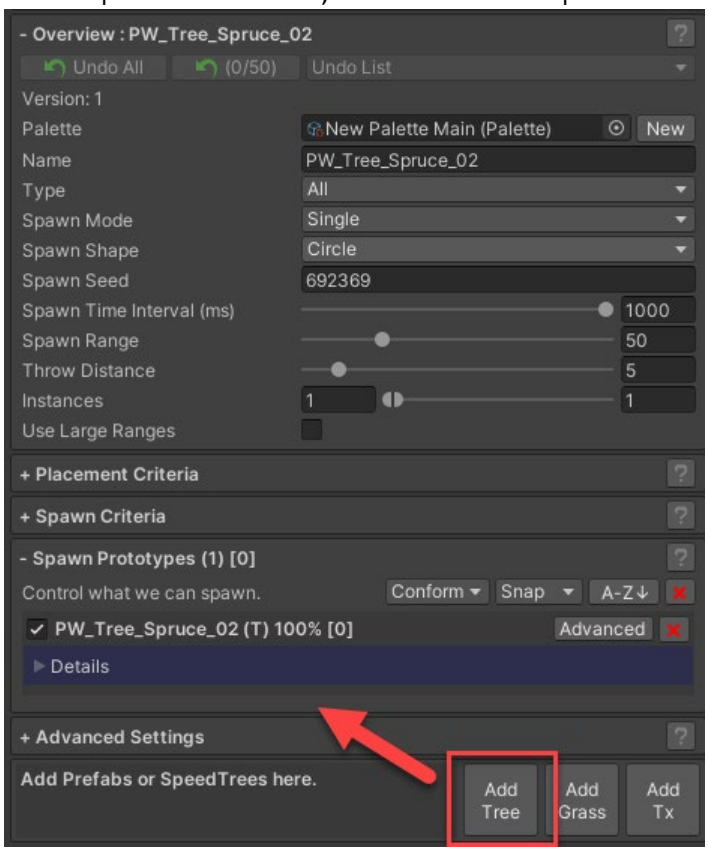
grass in the inverted section. The overall effect of this is a nice grassy field, and the interesting aspect of this is that you will get nice patches of alternating grasses as well. This tip applies to all types of spawning.

Spawning Terrain Trees

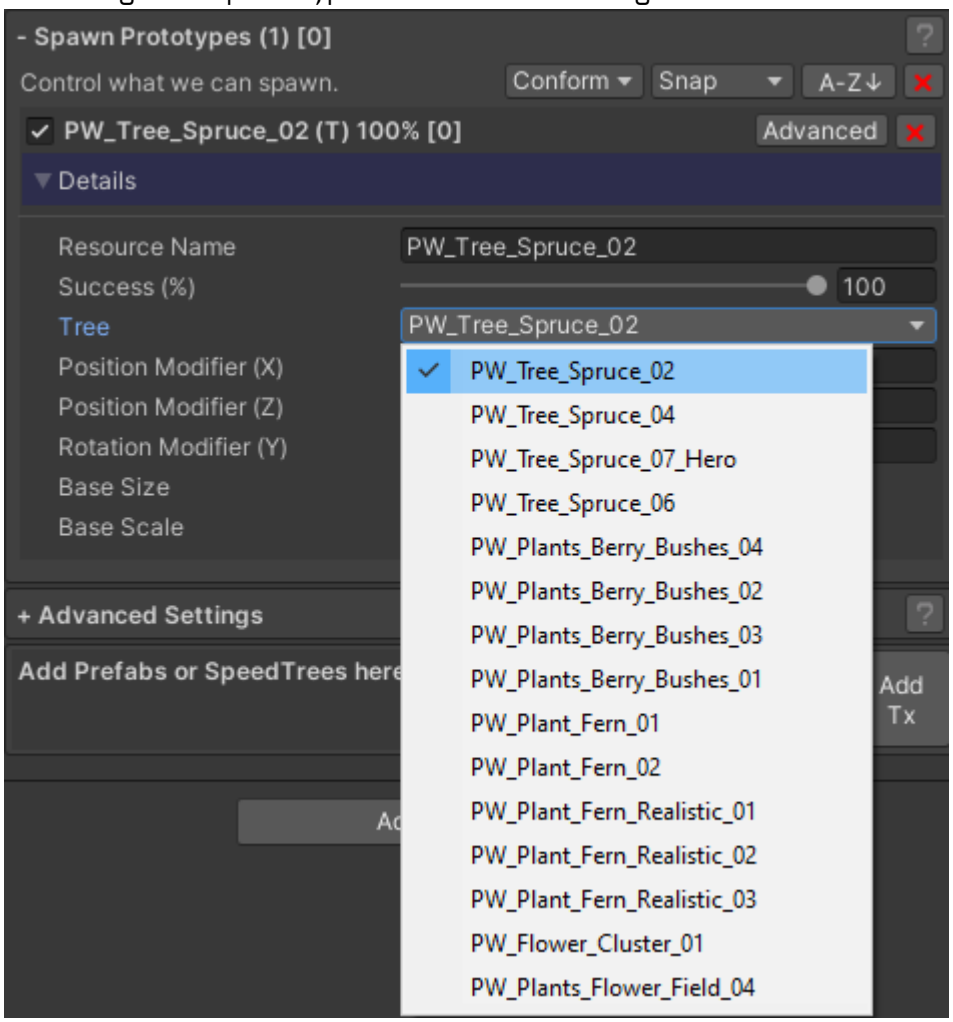
1. Add your trees to your terrain.



2. Add a spawner and add your trees to the spawner.



3. Go through each prototype and make sure the right tree is selected.

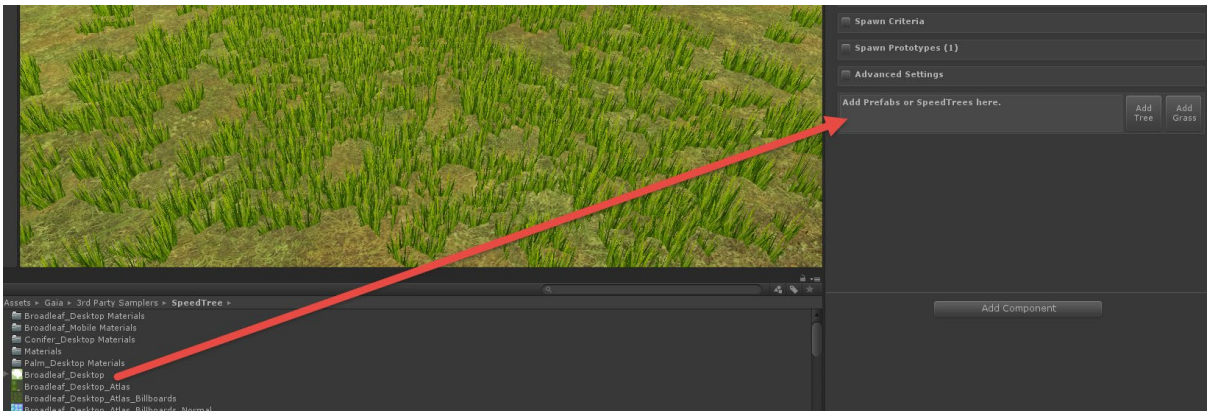


4. Spawn as usual.

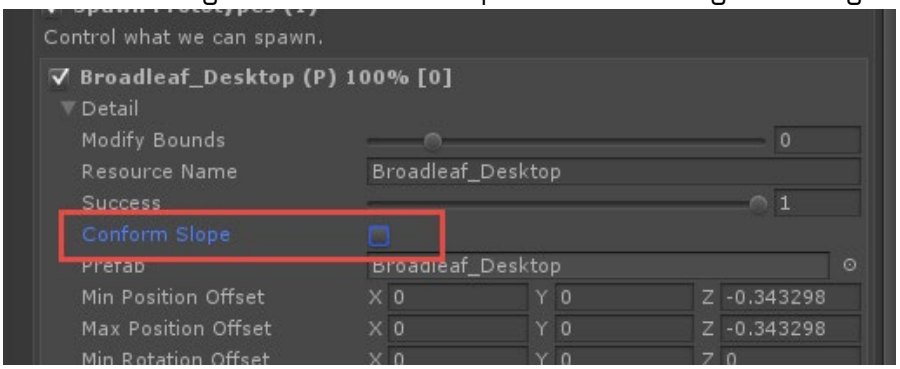
TIP: The noise generator generates noise in world space, and this means that multiple spawners can use it and expect it will generate the same mask. Try setting up two spawners with the same noise settings, and one, select Invert Mask. This will invert the generated pattern and you can use this to spawn one tree in one section, and another tree in the inverted section. The overall effect of this is a nice populated terrain, and the interesting aspect of this is that you will get nice patches of alternating trees as well. This tip applies to all types of spawning.

Spawning SpeedTrees As Prefabs

1. Add a spawner and drag and drop SpeedTree .spm file onto the spawner.



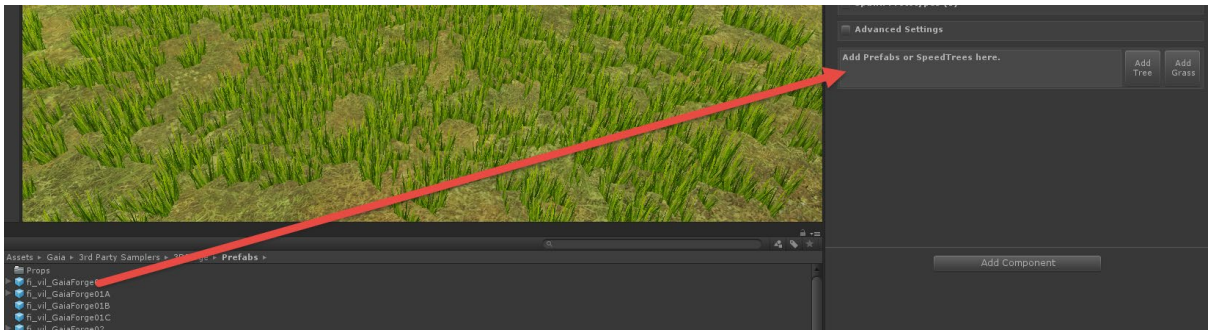
2. Consider turning off Conform To Slope as most trees grow straight up.



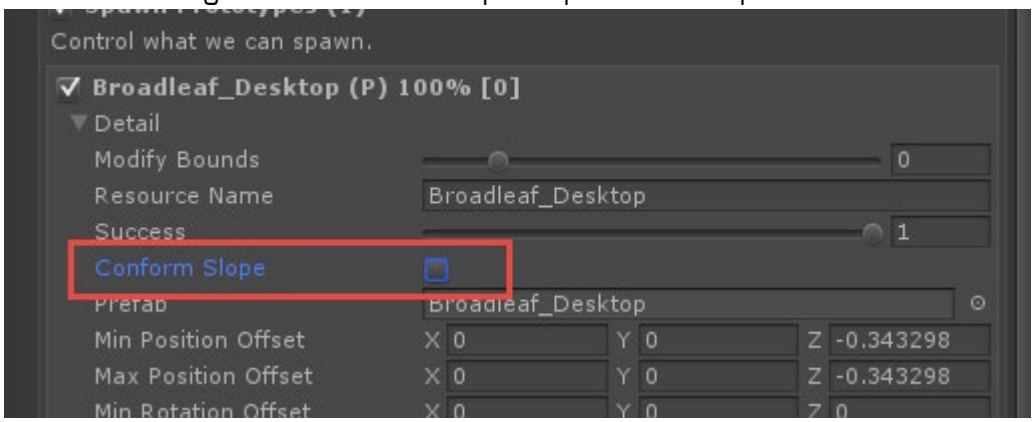
3. Spawn as usual.

Spawning Prefabs

1. Add a spawner and drag and drop a prefab onto the spawner.



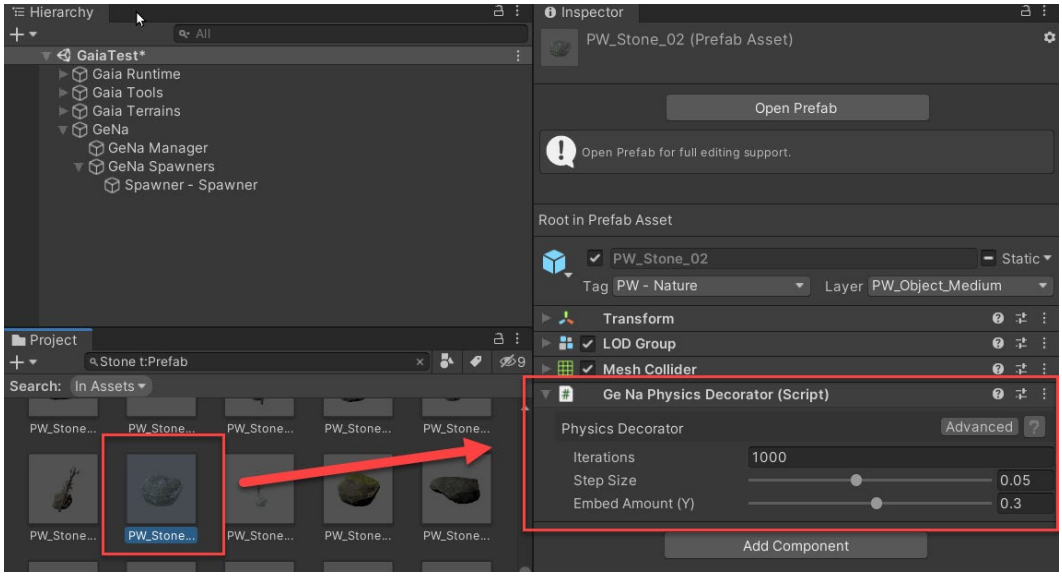
2. Consider turning off Conform To Slope. Depends on the prefab.



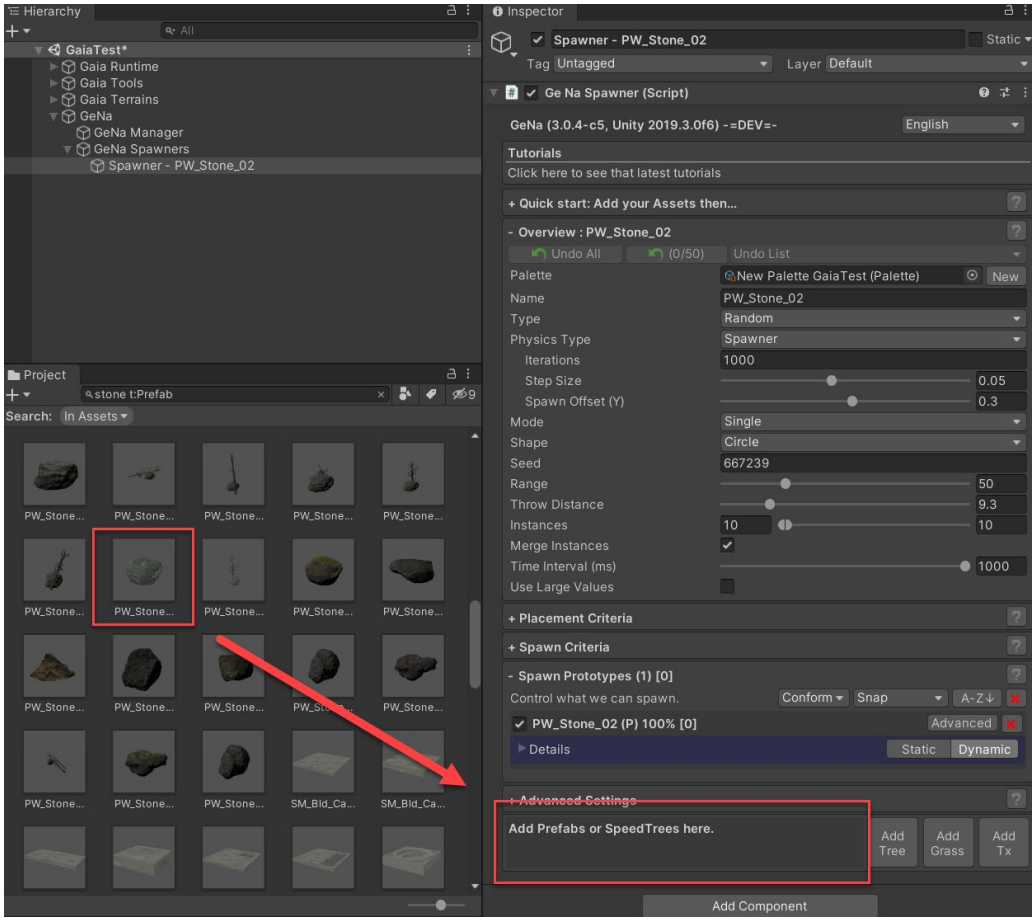
3. Spawn as usual.

Spawning with Physics (Gravity)

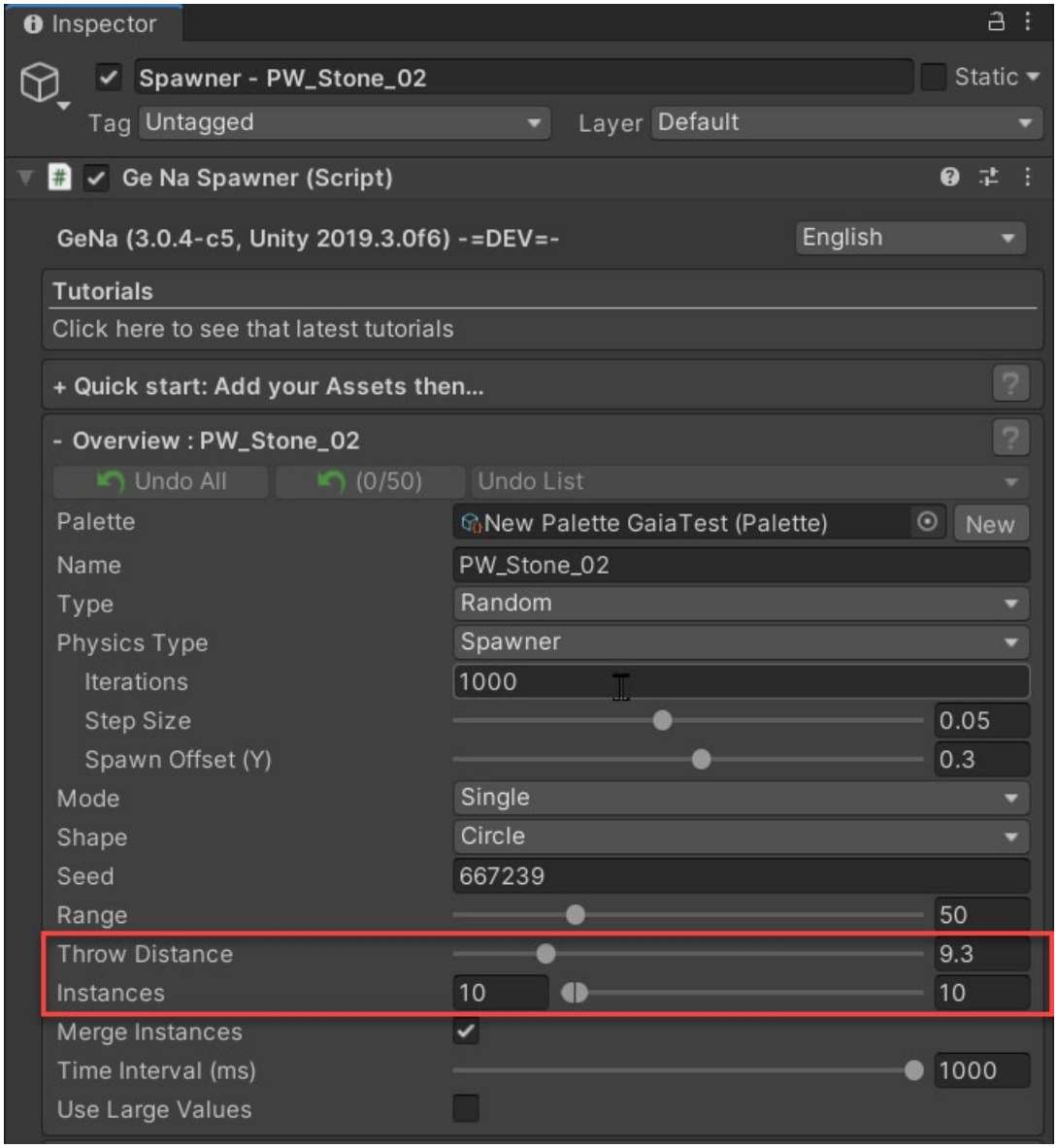
Step 1). **Select the Prefab** you wish to spawn and add a **Physics Decorator** to it.



Step 2). Ingest the selected prefab into a Spawner.

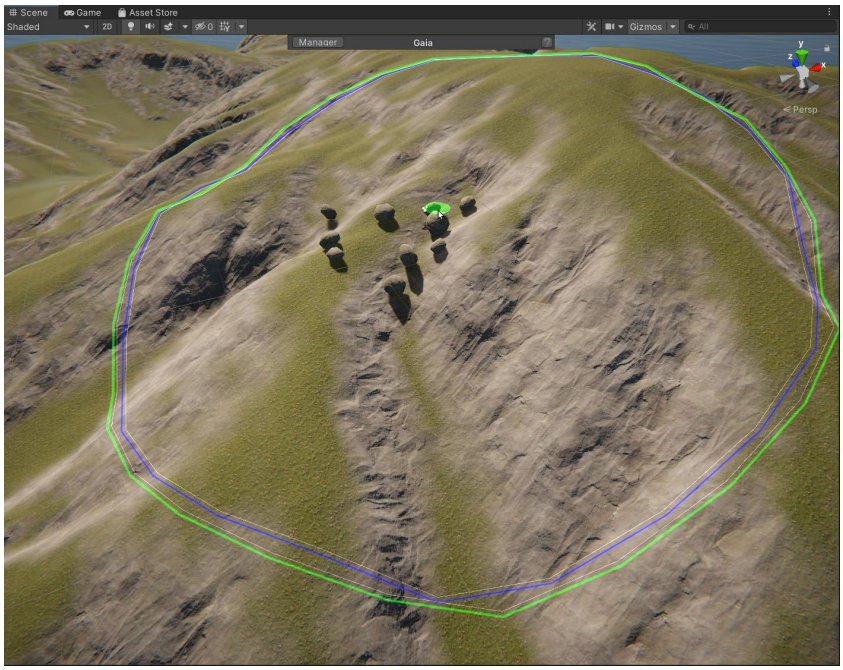


Step 3). Modify the Spawn Settings. In this case, we are going to increase the number of Instances and increase the Throw Distance.



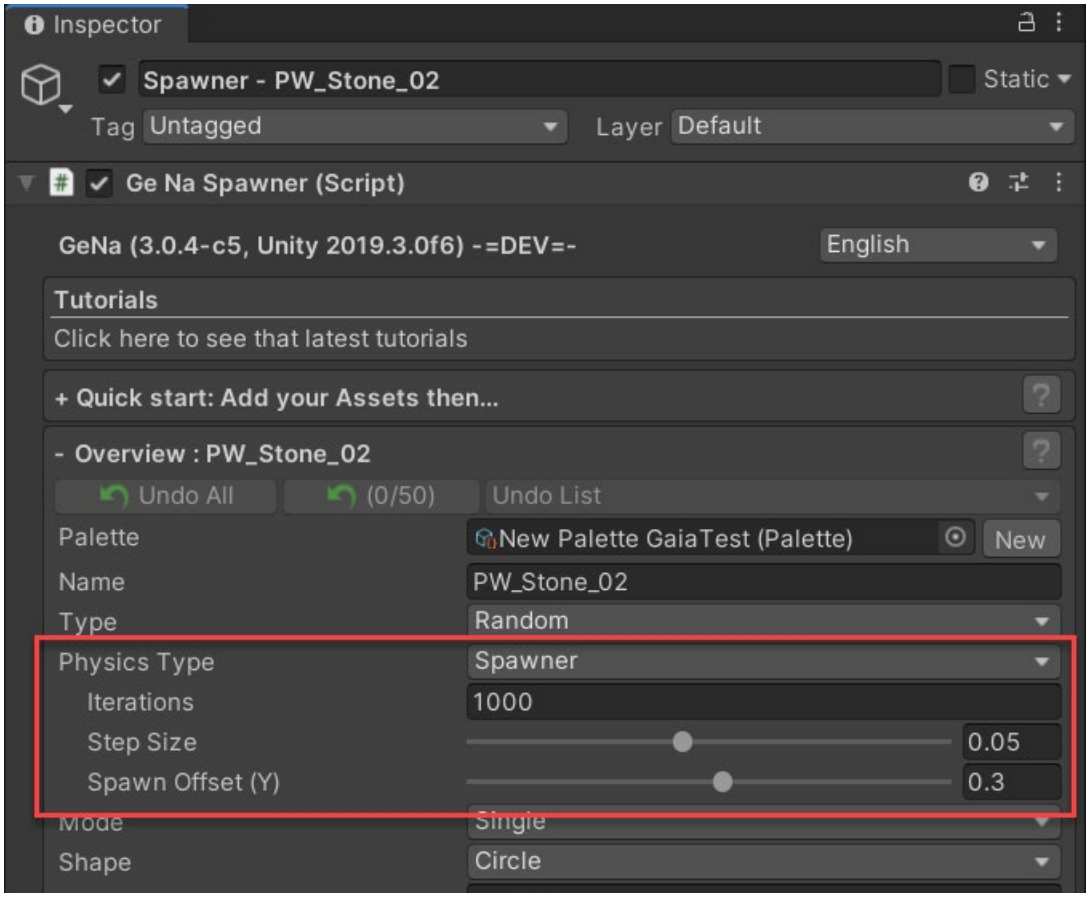
Step 4). Spawn the Prototype.

Performing a Spawn with this Decorator will allow objects to interact with the Environment.



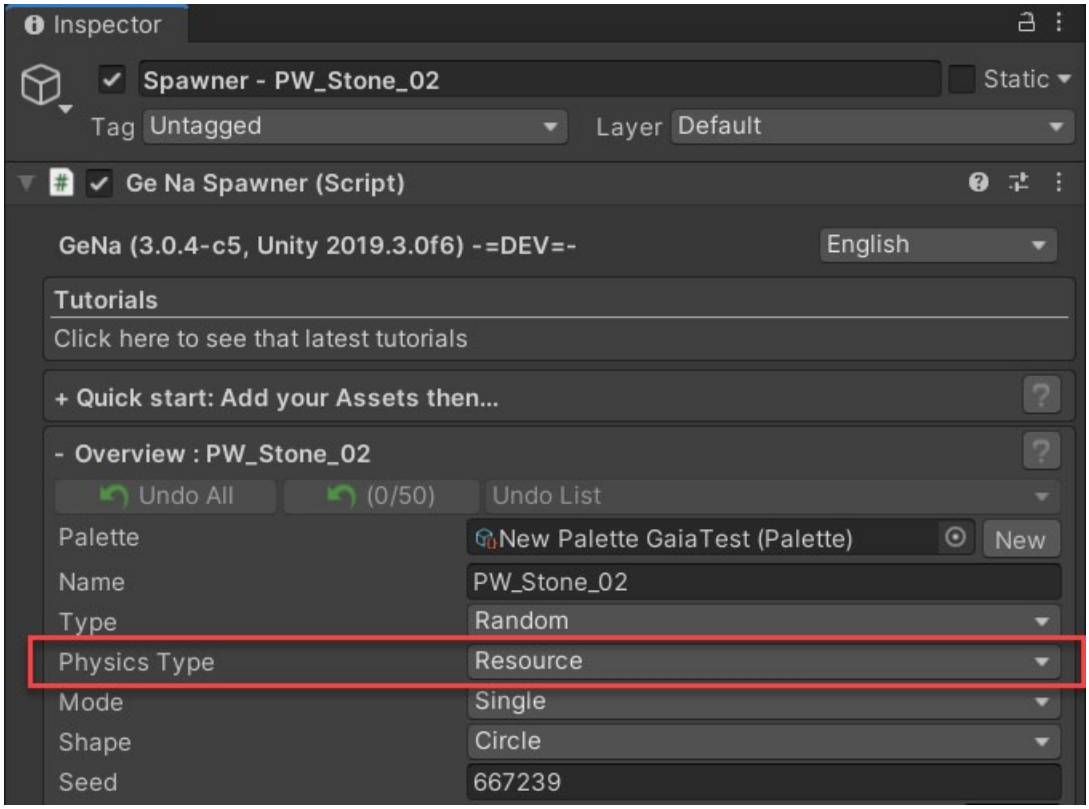
Step 5). There are two types of Spawn Modes – **Spawner** and **Resource**.

Spawner will spawn an object using the settings at the root level.

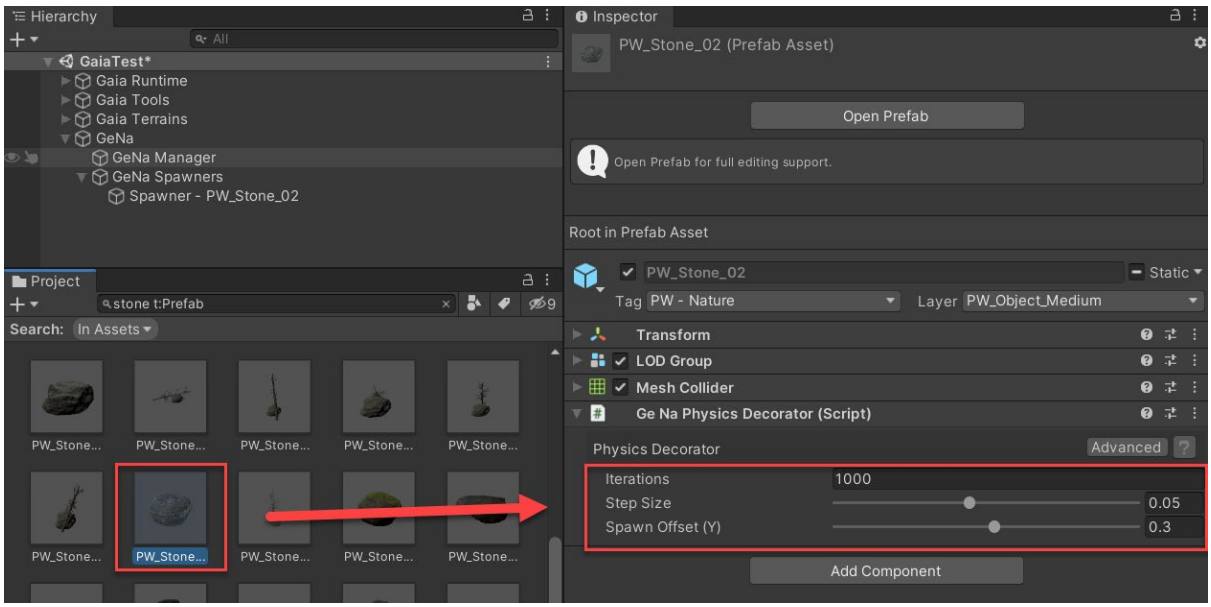


Resource will spawn an object using the Physics settings on the Decorator**.**

Set the **Physics Type** to '**Resource**'



And you can modify the Physics Settings in the Prefab itself.



Spawning Structures

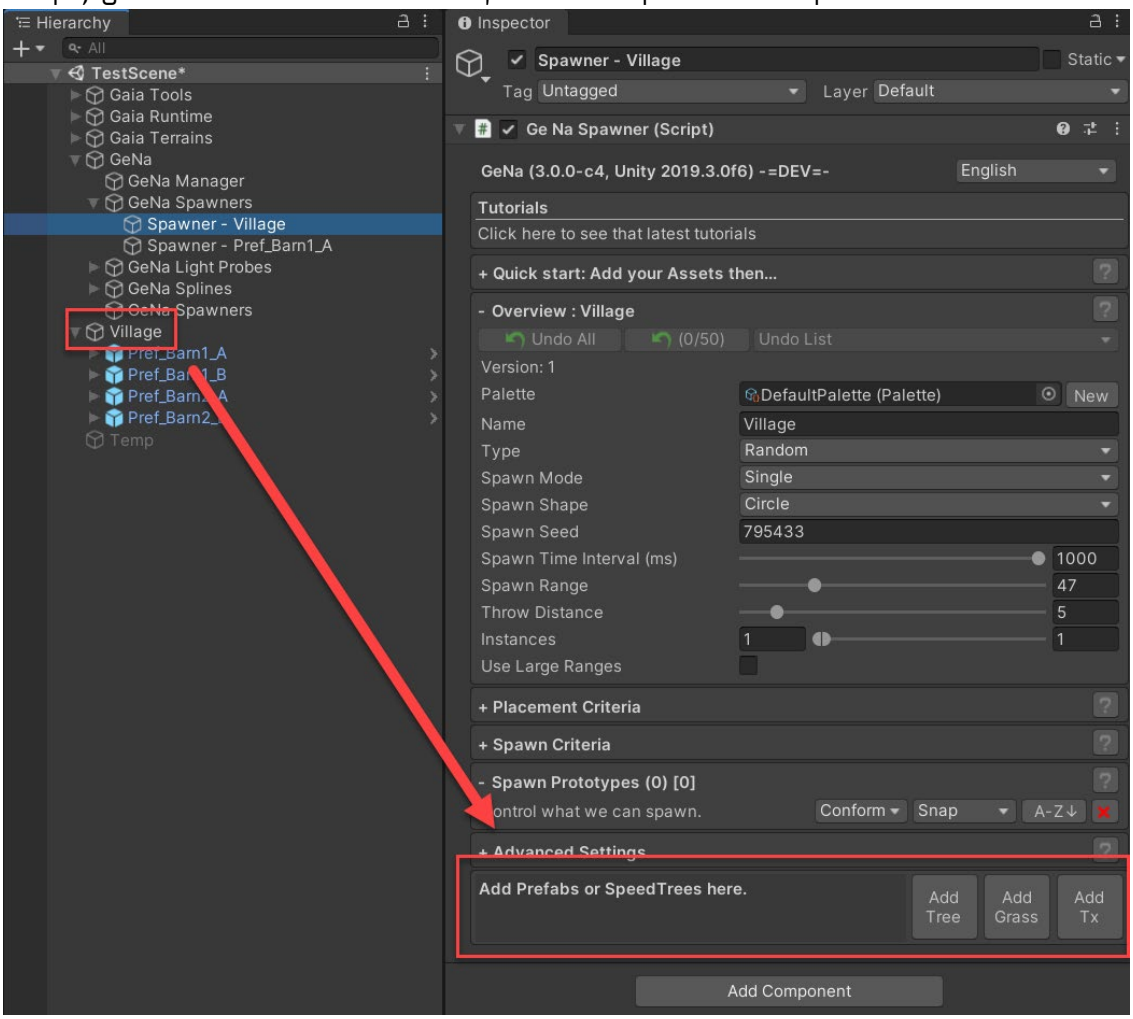
Sometimes you might want to setup a small village, town or just a collection of Game Objects that you have laid out in your Hierarchy. We refer to these in GeNa as 'Structures'.

To make Structured Spawners, all you need to do is **Drag the Root of your Structure into a GeNa Spawner** and GeNa will maintain the structure.

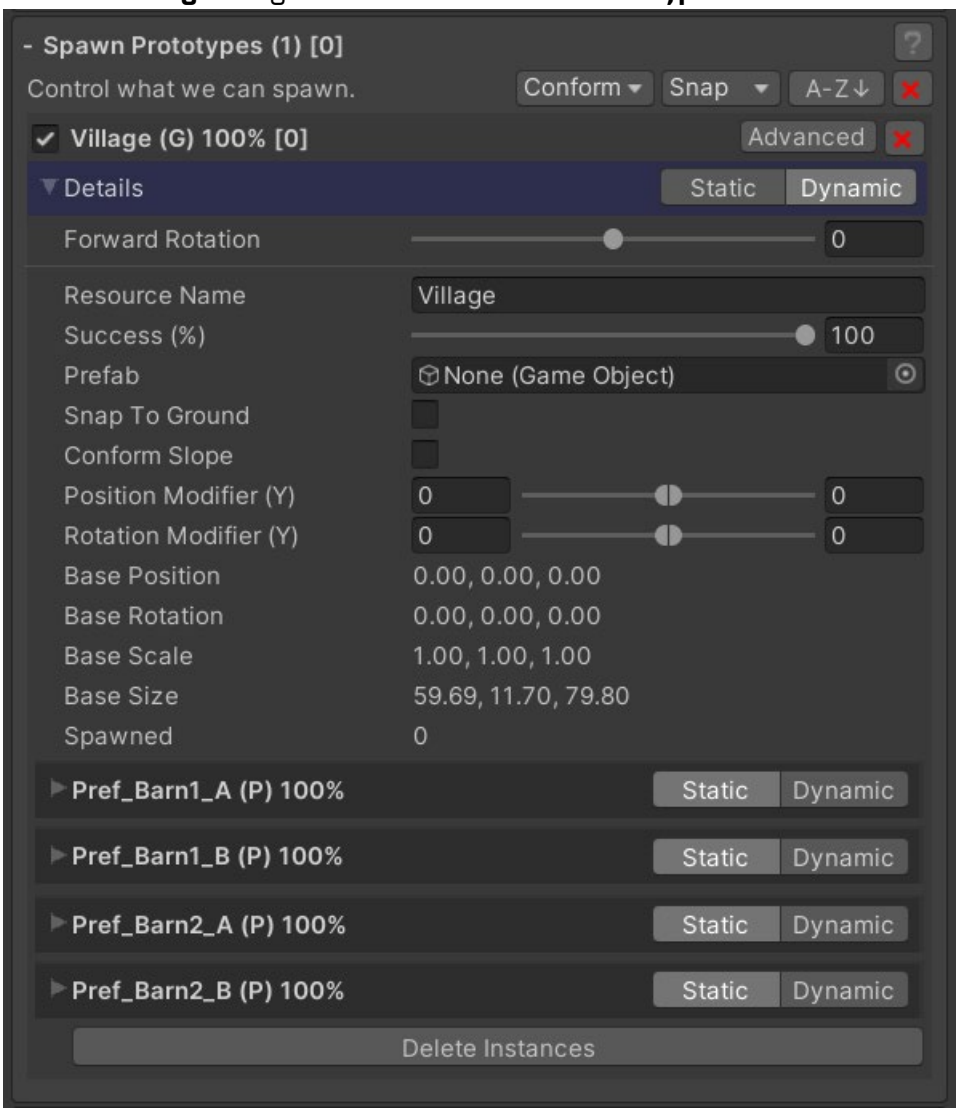
Here is an example:

1. Here we have a **Village Structure** along with the setup in the hierarchy

2. Simply grab the **Root** of the GameObject and Drop it into the Spawner



3. Now the **Village** is ingested as a **Structured Prototype**



When we perform a **Spawn**, the structure will be maintained.

Using Decorators

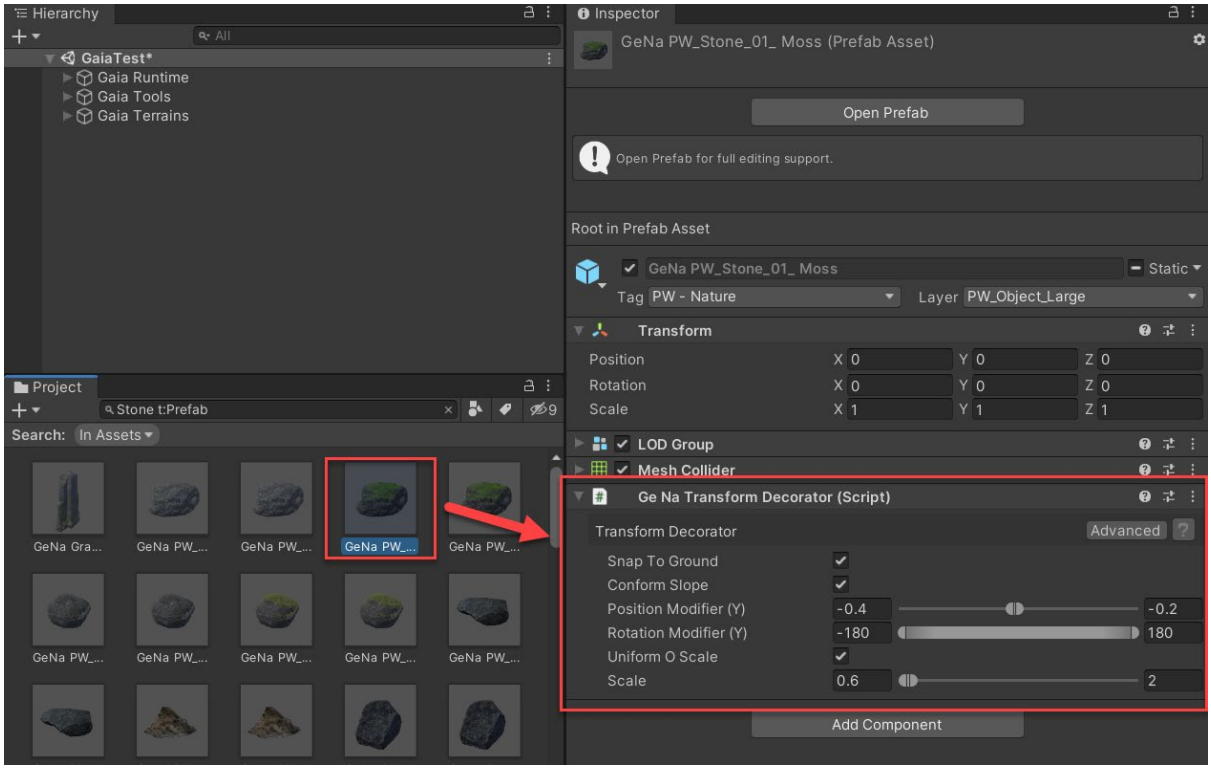
In GeNa, Decorators allow you to add more sophisticated logic to your Spawners.

For more information, see GeNa Decorators.

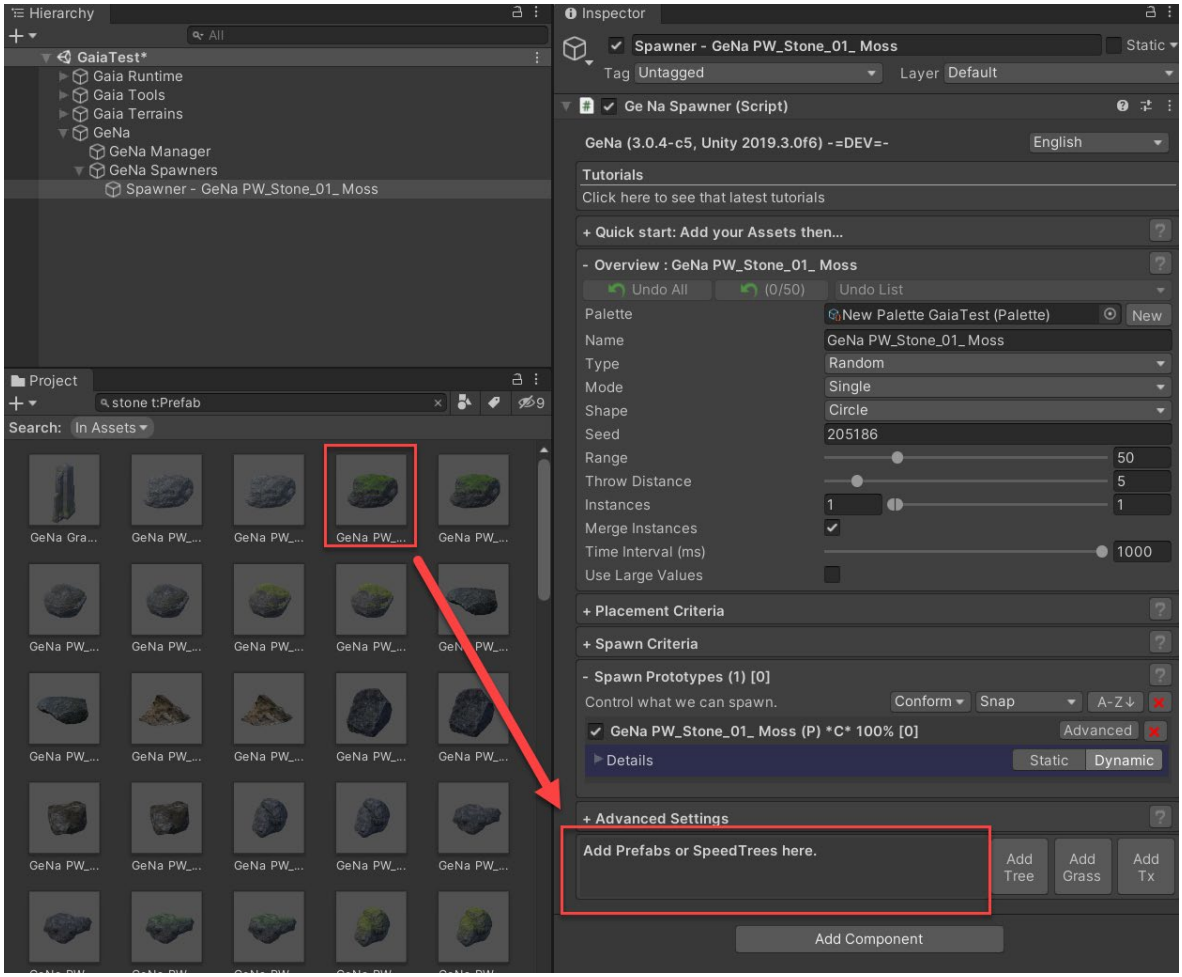
The following steps will guide you through the process of adding and using decorators in your GeNa Spawners.

Step 1). Add a GeNa Decorator to a Prefab.

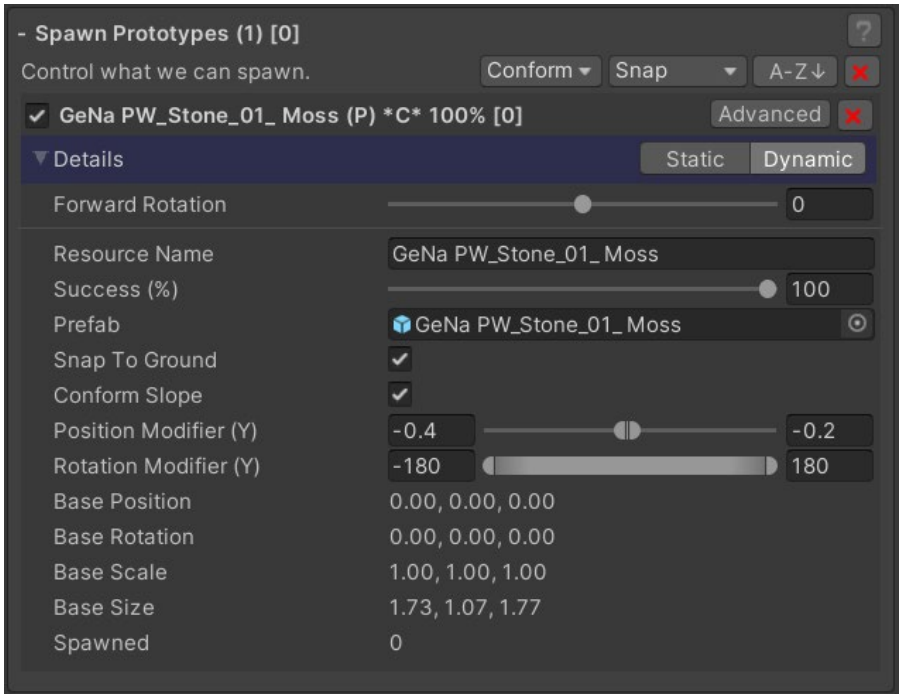
In this case, we will be adding the '**GeNa Transform Decorator**' to our **Rock**.



Step 2). Add the Prefab to a Spawner.



If you open the Prototype, you will notice that GeNa has modified the transform data with what was in the GeNa Transform Decorator.



Using Prefab Unpacker Decorator

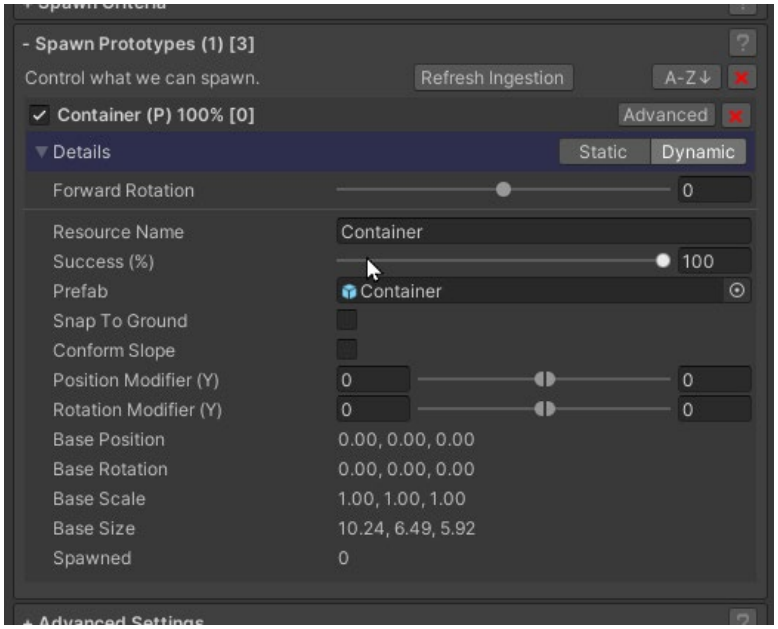
In order to setup decorators correctly, it's important to understand the structure of Unity's new prefab workflow system. See Unity's 'Prefab workflow' page for more information.

Unfortunately, GeNa won't know the contents of a prefab until it is 'unpacked' first. See Unity's 'Unpacking Prefab instances' page for more information.

Due to the nature of how this is designed to work, it's important that we follow a structured workflow when designing our spawn prototypes.

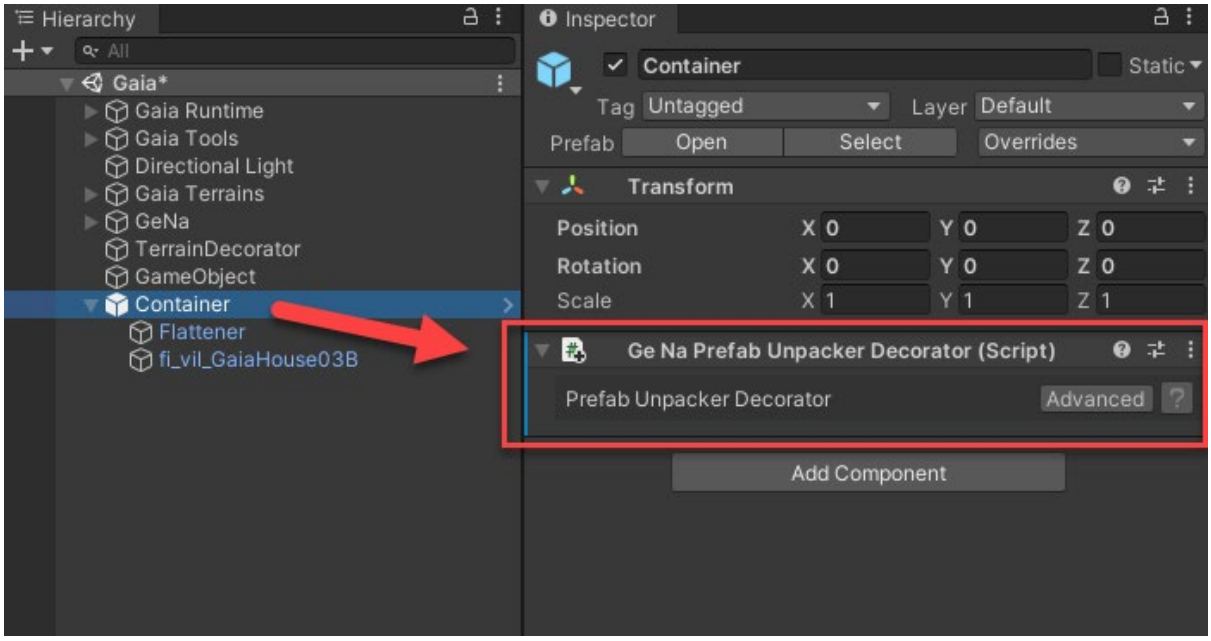
Let's say we want to spawn a house and have it flattened underneath. Typically, you would think to set it up like this.

Here we have a prefab called 'Container' that contains a Flattener and an **instance** of a house. When we ingest this container into a GeNa Spawner as-is, you'll notice that GeNa has added the following spawn prototype.

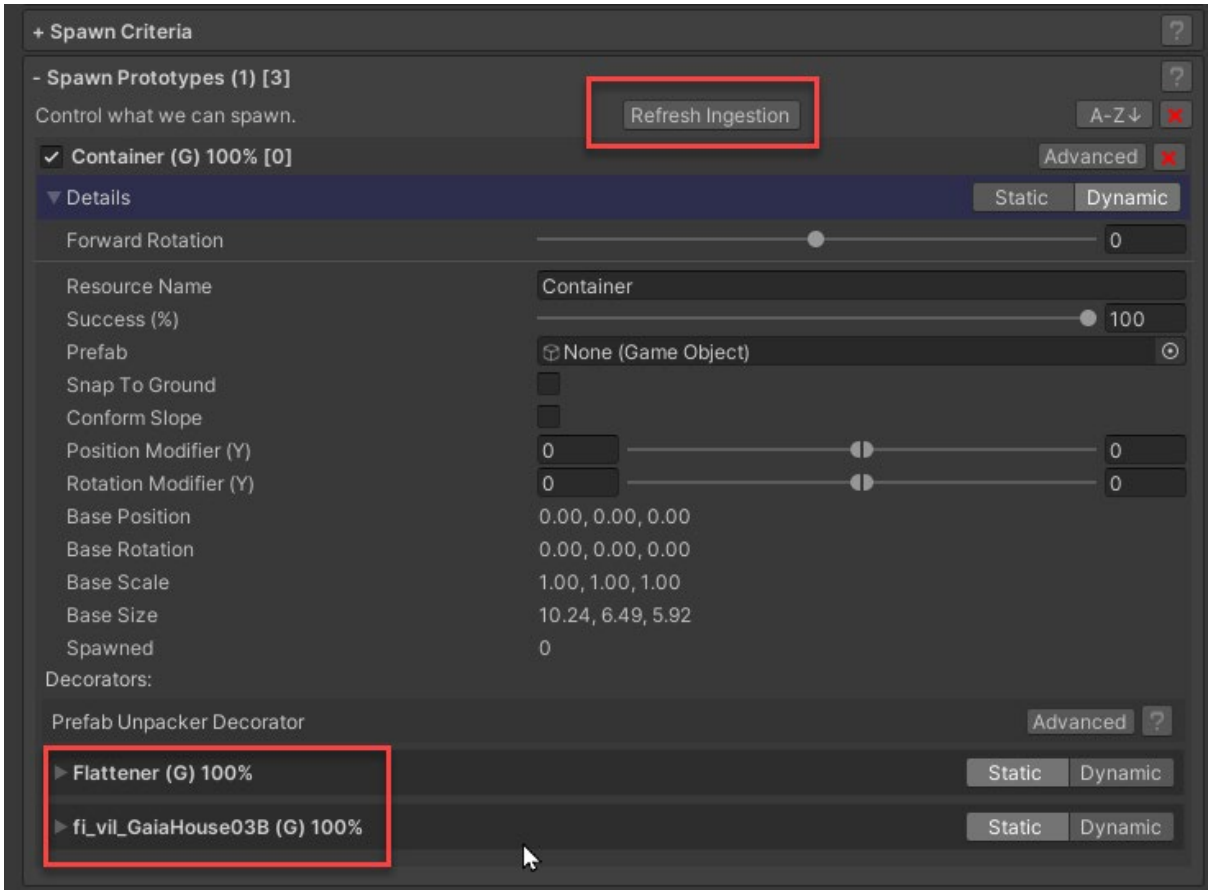


The problem is that when we spawn this object, GeNa doesn't know anything about the Flattener Decorator nested in the prefab.

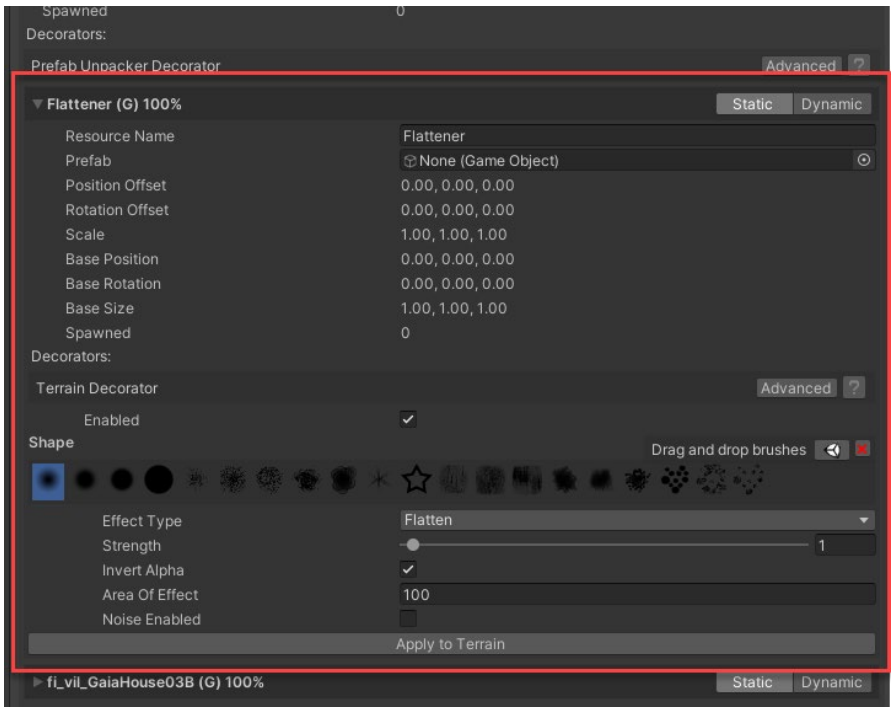
This means that the GameObject will need to be unpacked, therefore we add a '**GeNa Prefab Unpacker Decorator**' to the root of the prefab.



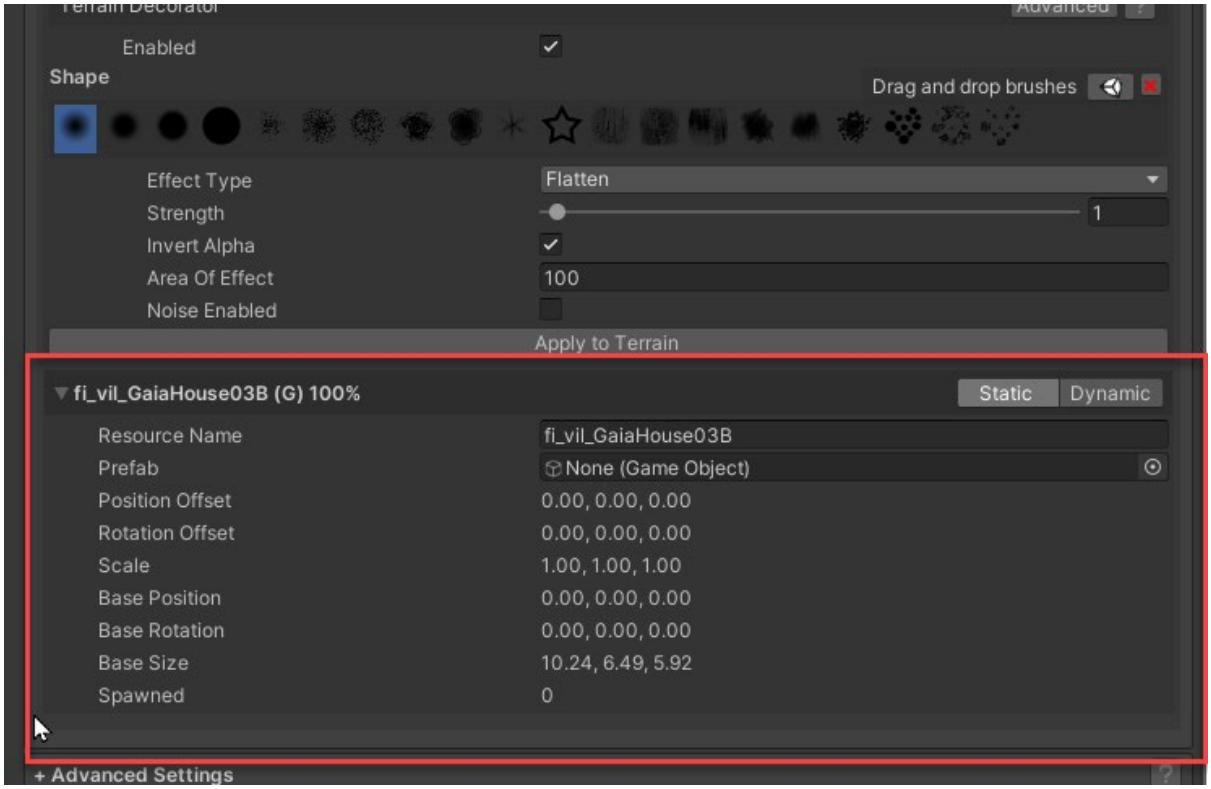
Once you apply this change to the prefab, you can now 'Refresh the Ingestion' and you'll see that GeNa has picked up the Flattener and the house instance.



If we look closely at the nested resource, we'll find the Terrain Decorator that was attached to this GameObject.

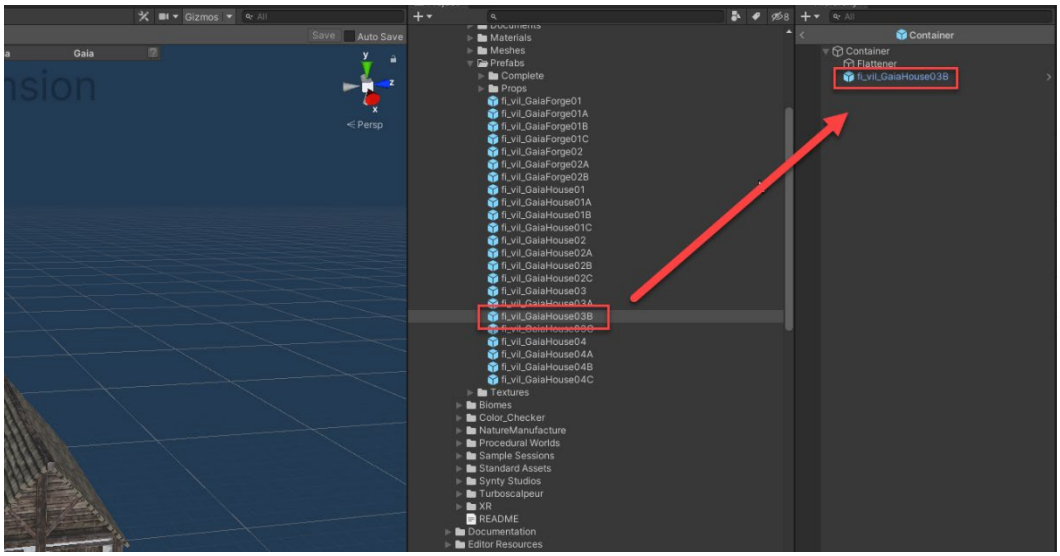


However, if we look at the **instance** of the house, it won't show up as a prefab in the resource and hence, will not spawn the GameObject.

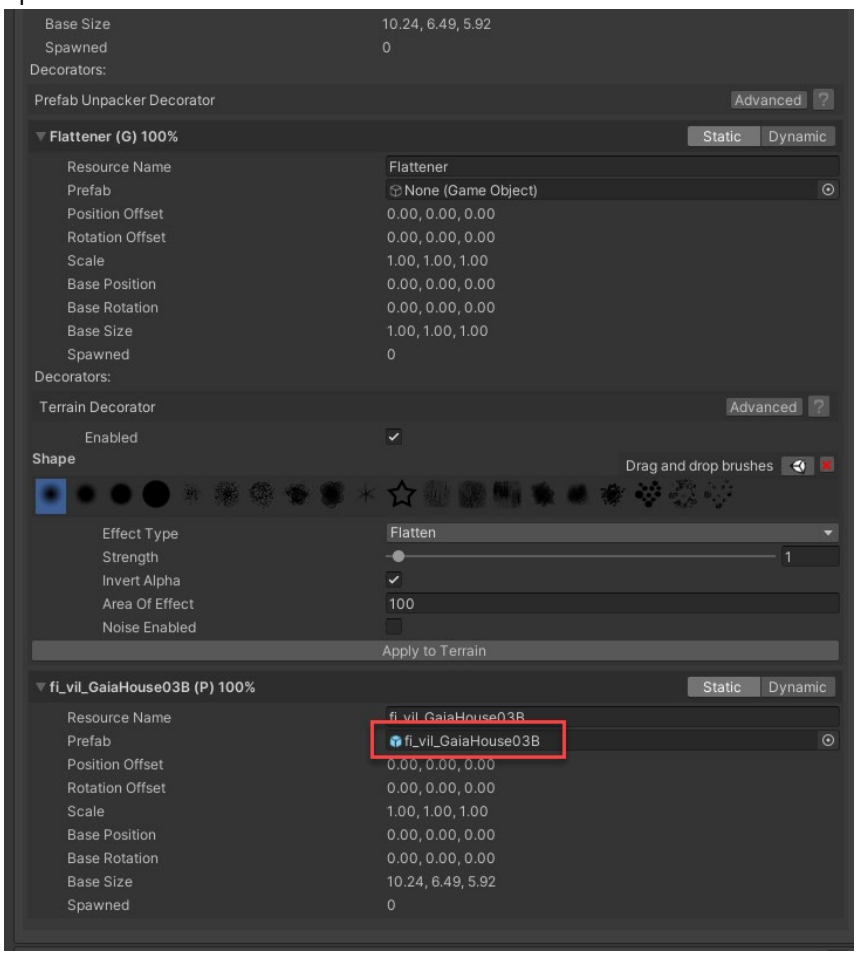


This is normal behaviour because GeNa does not know what the prefab of that object is and cannot store and spawn instances of that GameObject.

Therefore, to fix this issue we would need to ensure that the container has a reference to the prefab nested under the GameObject like so.



Now when you refresh the ingestion of the Spawner, GeNa will know what GameObject to spawn and it will be added to the Palette.



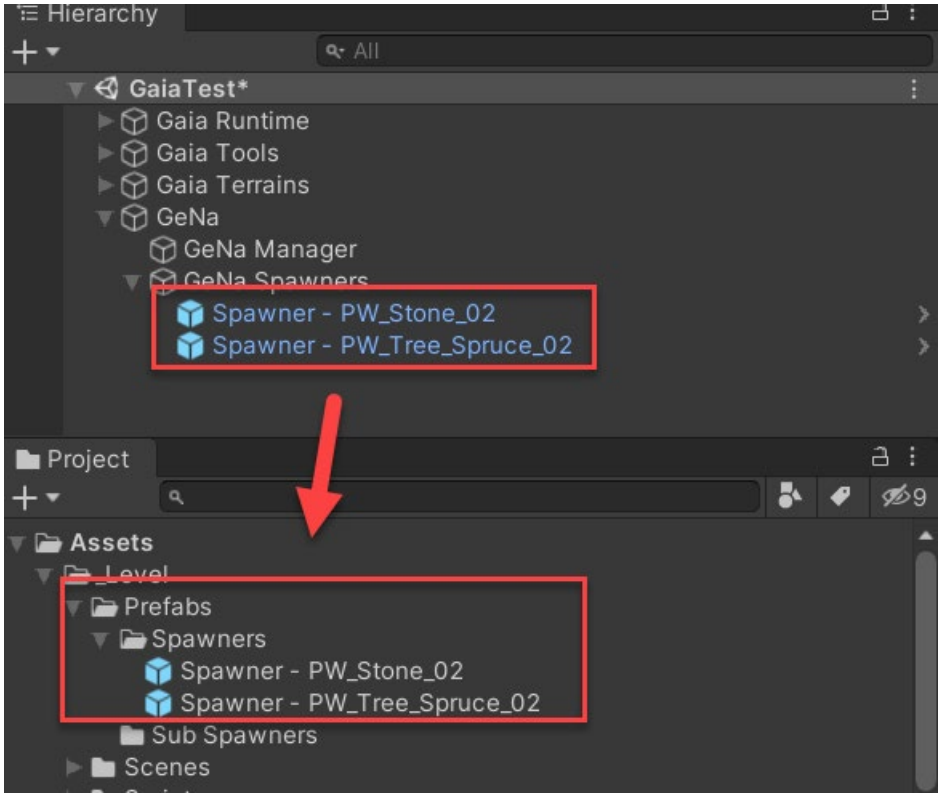
Using Sub Spawners

Gena has the ability run multiple spawns at one time as a single entity. This is great for things that involve multiple objects that can be logically spawned as one entity.

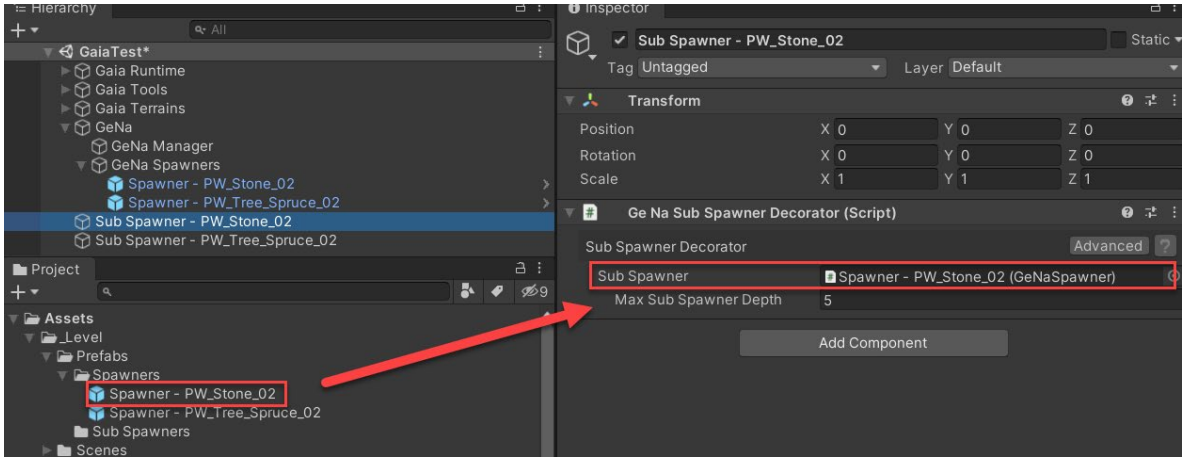
To make a subspawn, follow these steps:

Step 1). Create **Spawner Prefabs**

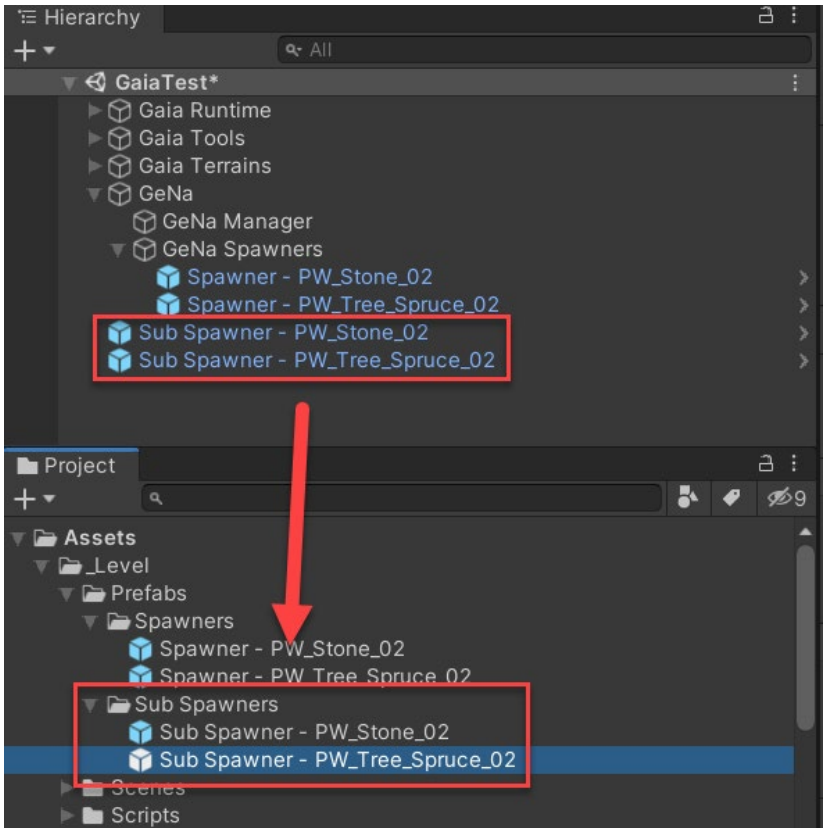
In order to perform Sub-Spawns, the Spawners need to be Prefabs. For example, we've created two different kind of Spawners – One that spawns Trees and one that Spawns Rocks with Physics. For more information, see Spawning with Physics.



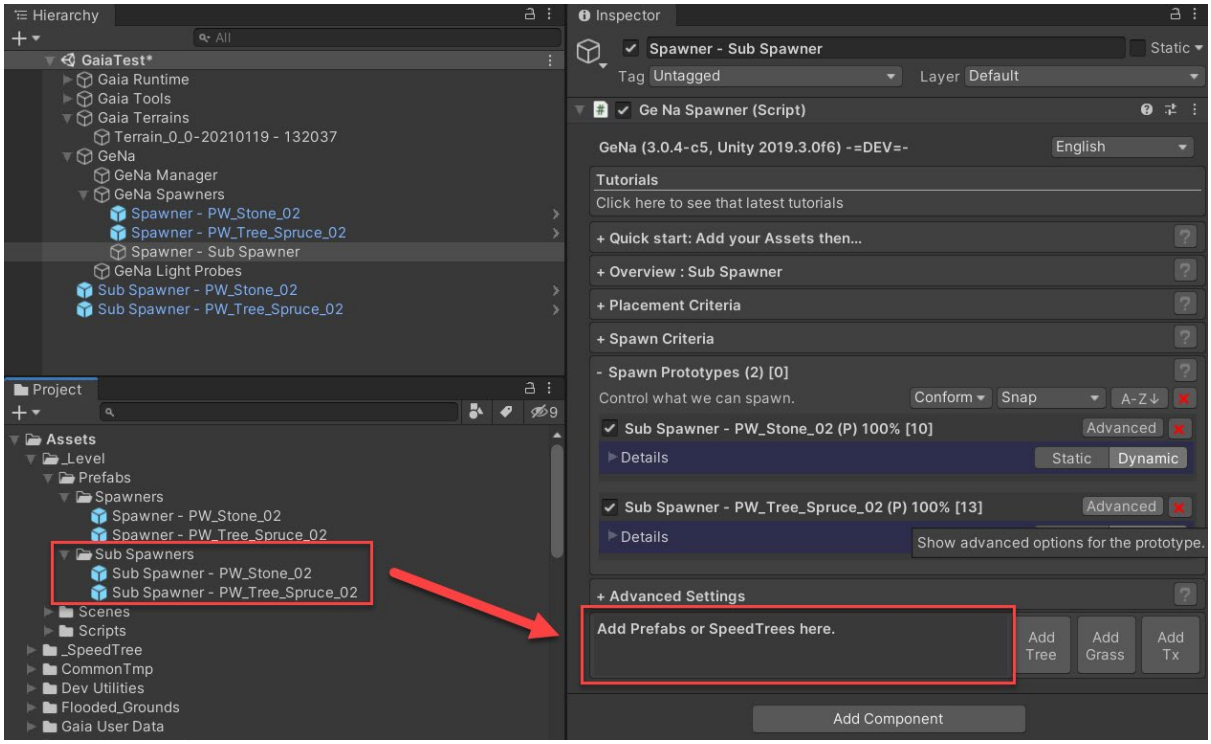
Step 2). Create Empty GameObjects and attach a '**GeNa Sub Spawner Decorator**' onto it. Then reference them each.



Step 3). Make prefabs out of the Sub Spawner objects.



Step 4). Attach the Sub Spawners to a new GeNa Spawner.


Hierarchy
GaiaTest*
Gaia Runtime
Gaia Tools
Gaia Terrains
Terrain_0_0-20210119 - 132037
GeNa
GeNa Manager
GeNa Spawners
Spawner - PW_Stone_02
Spawner - PW_Tree_Spruce_02
Spawner - Sub Spawner
GeNa Light Probes
Sub Spawner - PW_Stone_02
Sub Spawner - PW_Tree_Spruce_02
Project
Assets
_Level
Prefabs
Spawners
Spawner - PW_Stone_02
Spawner - PW_Tree_Spruce_02
Sub Spawners
Sub Spawner - PW_Stone_02
Sub Spawner - PW_Tree_Spruce_02
Scenes
Scripts
_SpeedTree
CommonTmp
Dev Utilities
Flooded_Grounds
Gaia User Data
Inspector
Spawner - Sub Spawner
Static
Tag Untagged
Layer Default
Ge Na Spawner (Script)
GeNa (3.0.4-c5, Unity 2019.3.0f6) -=DEV=-
English
Tutorials
Click here to see that latest tutorials
+ Quick start: Add your Assets then...
+ Overview : Sub Spawner
+ Placement Criteria
+ Spawn Criteria
- Spawn Prototypes (2) [0]
Control what we can spawn.
Conform
Snap
A-Z
Sub Spawner - PW_Stone_02 (P) 100% [10]
Advanced
Details
Static
Dynamic
Sub Spawner - PW_Tree_Spruce_02 (P) 100% [13]
Advanced
Details
Show advanced options for the prototype.
+ Advanced Settings
Add Prefabs or SpeedTrees here.
Add Tree
Add Grass
Add Tx
Add Component

Step 5). Perform a Spawn

There should be two spawners running simultaneously.

Mesh Based Workflow

Mesh based work flow – i.e. the process of spawning onto meshes is the same as the terrain based workflow with the exception that you cannot spawn terrain textures, grass or trees.

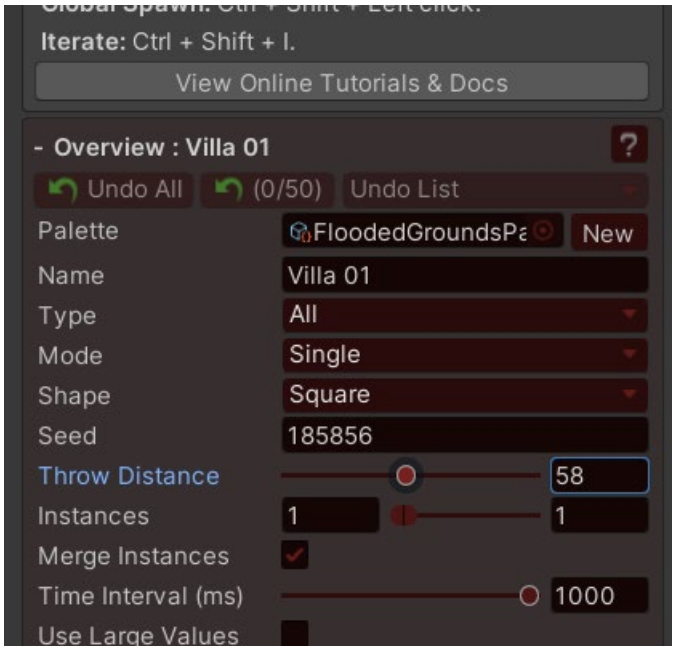
Set up your spawner and then shift click to visualise, and ctrl click to place. The target that you will spawn your prefabs on will be the mesh that you clicked on.

Prefabbing / Reuse Workflow

GeNa Spawners and their configurations can be saved as prefabs, and then re-instantiated and used like original objects. The benefit of this is that you can save your spawner configurations and re-use them later. Treat them as normal prefabs and the other operations and workflows described in this document will continue to work normal.

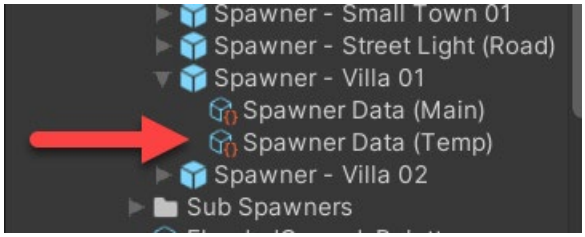
In order to support large amounts of prototypes and resources, GeNa Spawner Prefabs will automatically create a ScriptableObject asset inside of the Prefab in the Project window.

You'll also notice that if you change any property of an instance of your GeNa Spawner, the editor will turn red.

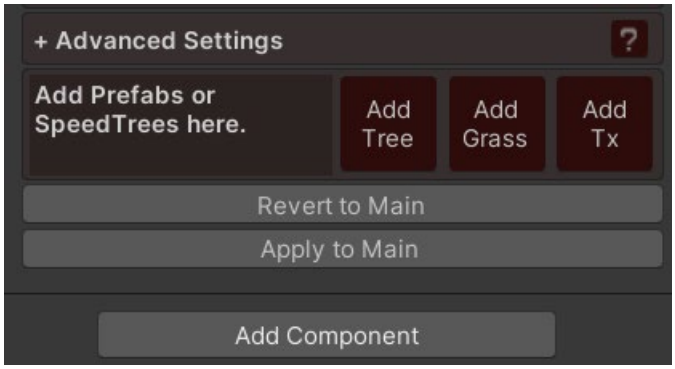


This is because the changes made are currently saved to the **Scene** and not the **Prefab** itself. This ensures that any changes you make will not overwrite your existing configuration.

GeNa does this by creating a separate '**Temp**' ScriptableObject *for each Instance* you modify inside of the Prefab itself.



If you scroll all the way to the bottom of the GeNa Spawner, you'll be prompted with two options:



If you wish to revert these settings back to the original configuration, click **Revert to Main.**If you wish to overwrite your prefab's settings with this scene instance, click **Apply to Main**.

Using the Animator Decorator

It's possible to perform animations upon spawning a Prefab in GeNa.

The Animator Decorator allows Animations to be performed when Spawning a Prefab.

Step 1). Attach a '**GeNaAnimatorDecorator**' to a Prefab that contains an Animator Component.

Step 2). Drag and Drop the Prefab into a Spawner.

Step 3). Spawn!

Using the Particle Decorator

It's possible to run Particle Emissions upon spawning a Prefab in GeNa.

The Particle Decorator allows Particle Systems to Emit when Spawning a Prefab.

Step 1). Attach a '**GeNaParticleDecorator**' to a Prefab that contains a Particle Systems Component.

Step 2). Drag and Drop the Prefab into a Spawner.

Step 3). Spawn!

Creating a Custom Decorator

It's possible to create your own *custom Decorators* in GeNa by creating your own Scripts that utilize GeNa's API. The following will demonstrate how to create a decorator that will **Scale objects when they Spawn**.

Step 1). Create a new Script titled '**GeNaGrowthDecorator**' somewhere in your project.

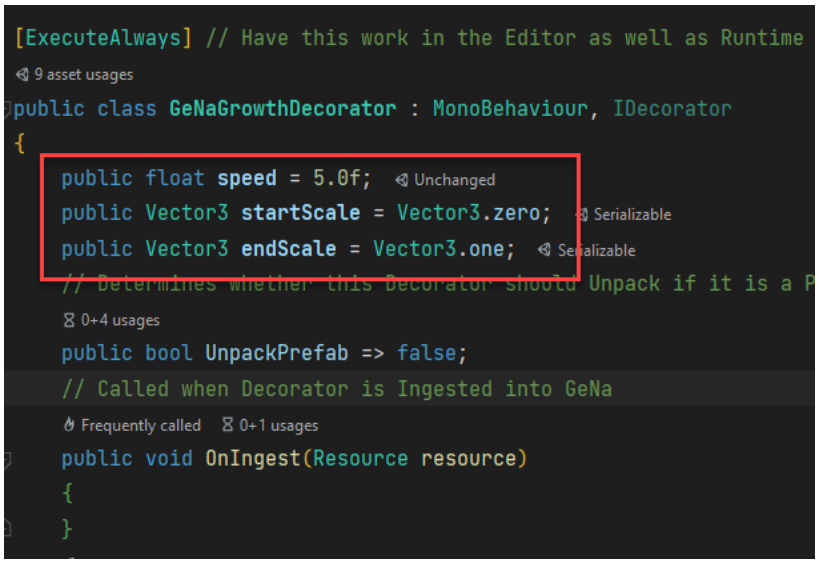
Step 2). Ensure you are using the **GeNa.Core** namespace.

Step 3). Inherit from the "**IDecorator**" interface and implement all the following default properties, functions/methods:

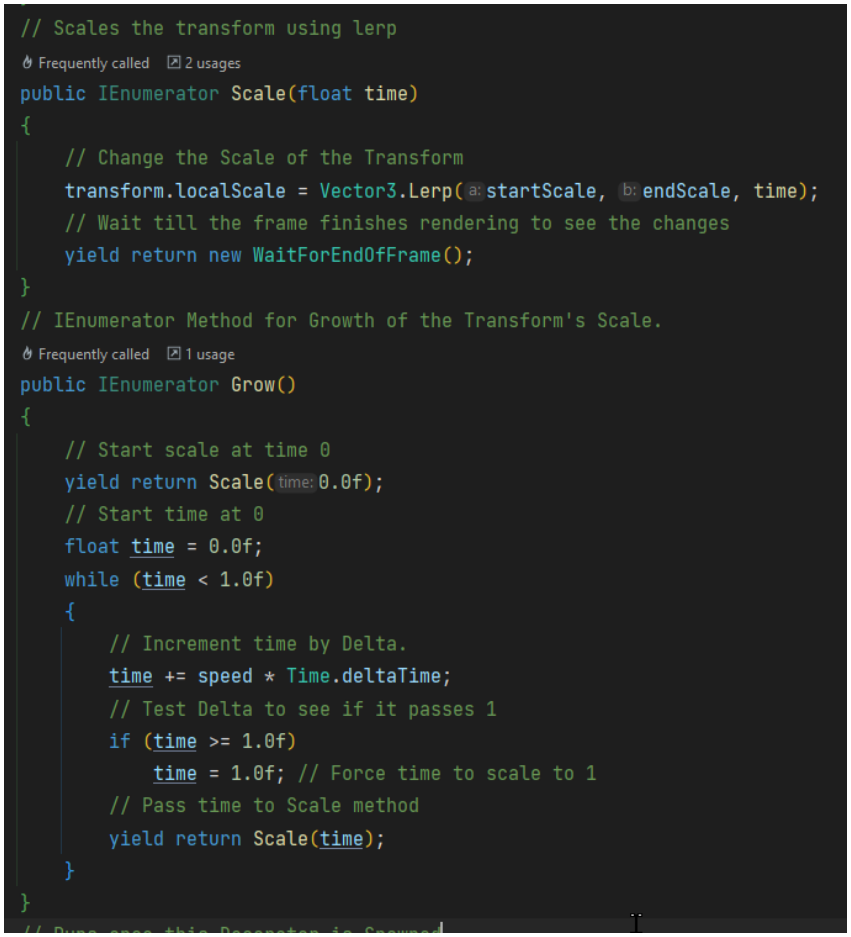
```
using System.Collections;
using UnityEngine;
using GeNa.Core;
// To make it work in the Editor as well as Runtime
[ExecuteAlways]
9 asset usages
public class GeNaGrowthDecorator : MonoBehaviour, IDecorator
{
    // Determines whether this Decorator should Unpack if it is a Prefab.
    0+4 usages
    public bool UnpackPrefab => false;
    // Called when Decorator is Ingested into GeNa
    Frequently called  0+1 usages
    public void OnIngest(Resource resource)
    {
    }
    // Runs once this Decorator is Spawned
    Frequently called  0+1 usages
    public IEnumerator OnSelfSpawned(Resource resource)
    {
        // Skips a frame
        // yield return null;
        // Continues to next process without skipping a frame
        yield break;
    }
    // Runs directly after Spawning Children Decorators
    Frequently called  0+1 usages
    public void OnChildrenSpawned(Resource resource)
    {
    }
    // Runs when Spawner has requested to load references from a Palette
    Frequently called  0+1 usages
    public void LoadReferences(Palette palette)
    {
    }
}
```

To understand more about when these methods are called, please refer to the comments in the image above.

For this demo, we're going to create a few variables to control the Scaling.



We're also going to have an IEnumerator method called 'Grow()' that will use a Coroutine to increase the size of the Transform when you Spawn the object.



To spawn things Asyncronously, you can use the **GeNaEvents.StartCoroutine** method which performs a Coroutine in Edit Mode and Play Mode.

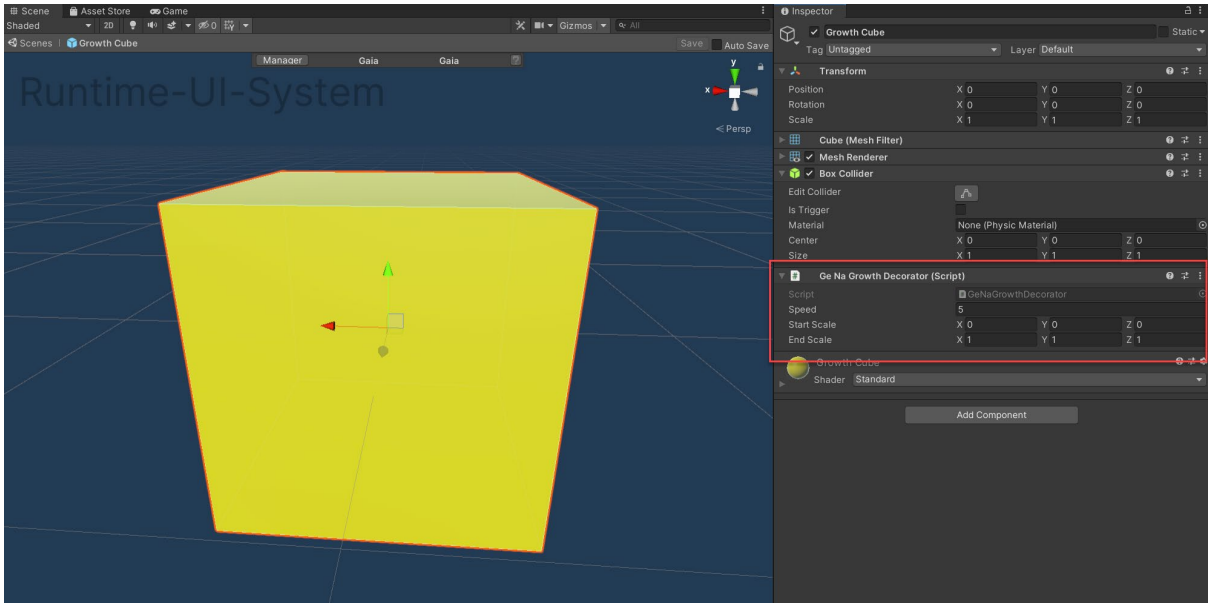
Or alternatively, you can use '**yield return Grow();**' to perform the operation Synchronously.

```
}
// Runs once this Decorator is Spawned
Frequently called  0+1 usages
public IEnumerator OnSelfSpawned(Resource resource)
{
    // Special method that performs a Coroutine at Edit mode and Runtime
    GeNaEvents.StartCoroutine(Grow(), this);
    // Alternatively, you can yield the Grow process directly (note: this halts spawn process in order to perform spawns).
    // yield return Grow();

    // Skips a frame
    // yield return null;
    // Continues to next process without skipping a frame
    yield break;
```

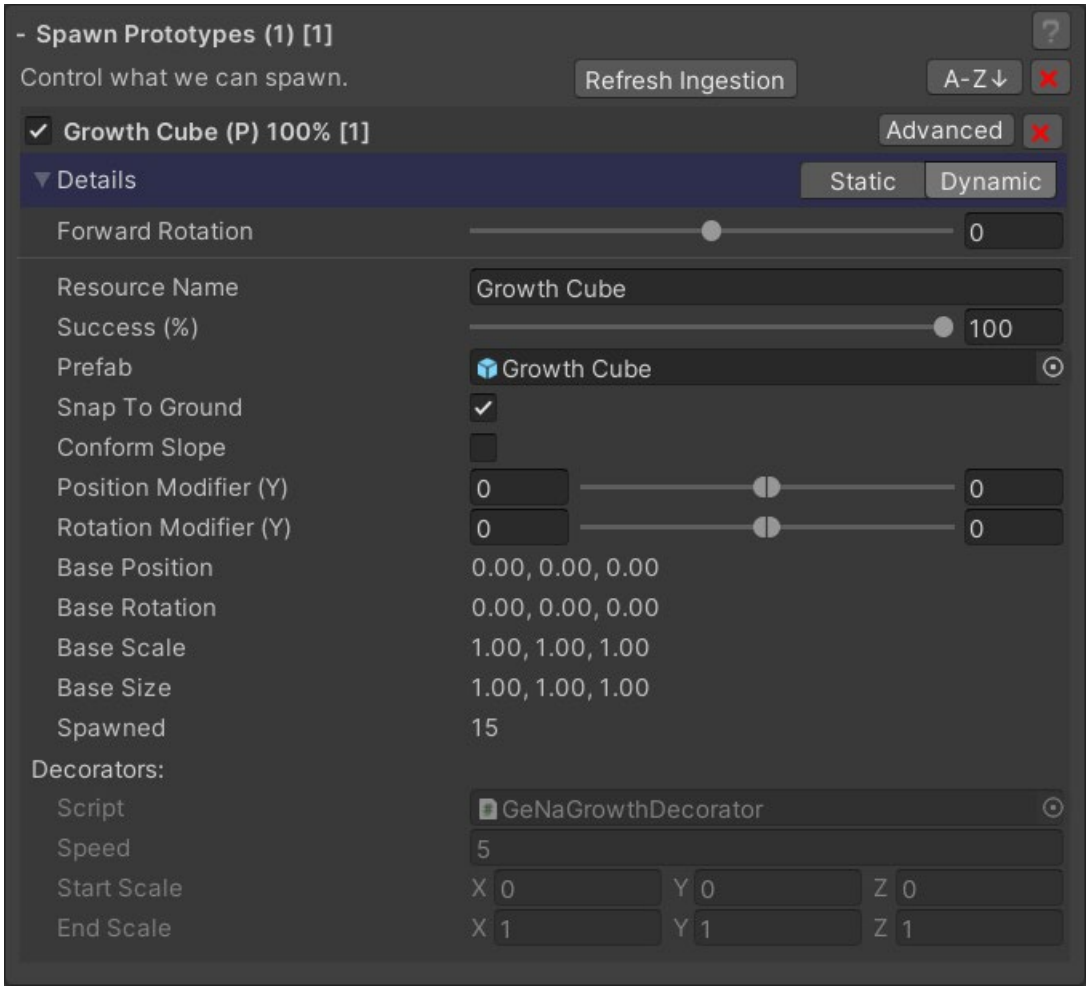
If the object gets destroyed as it's performing a Coroutine, it's important to tell Unity to stop any Coroutines running on this MonoBehaviour.

```
Event function
private void OnDestroy()
{
    StopAllCoroutines();
}
```

Step 4). Attach the GeNaGrowthDecorator to a Prefab.



Step 5). Drag and Drop the Prefab into a Spawner.

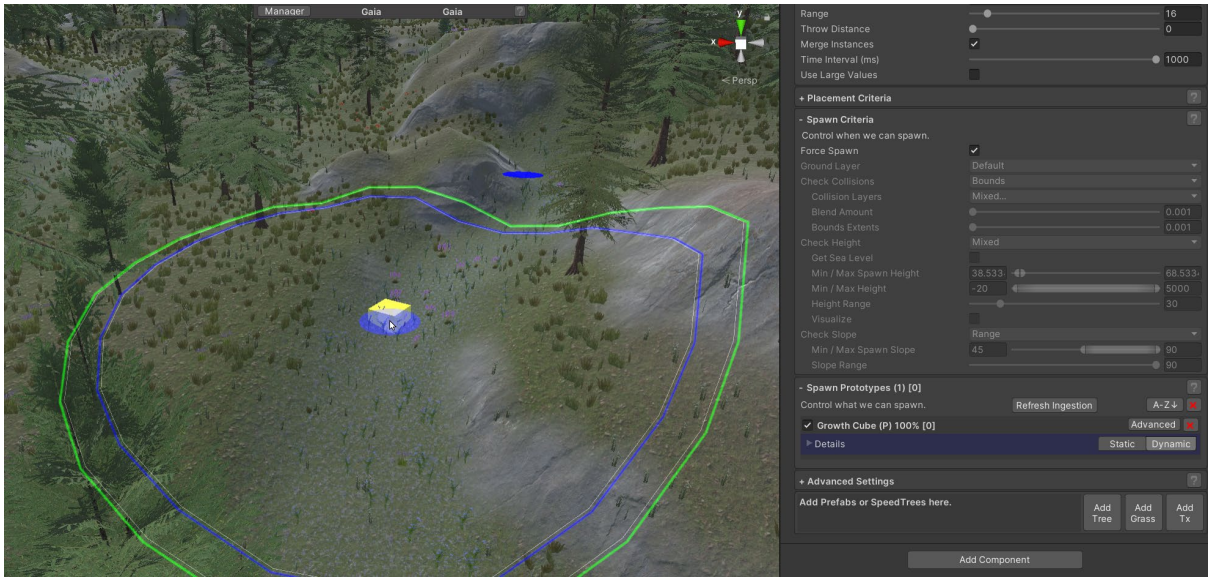


You'll notice that the Decorator isn't editable when it's attached to a spawner. The reason for this is because it's being rendered inside of a GeNa Spawner. In order to edit it, you'll need to make a new Editor Script for GeNaGrowthDecorator.

Example as follows. Be sure to place this script into an 'Editor' subdirectory. See Unity's Special Folders Documentation to find out more.

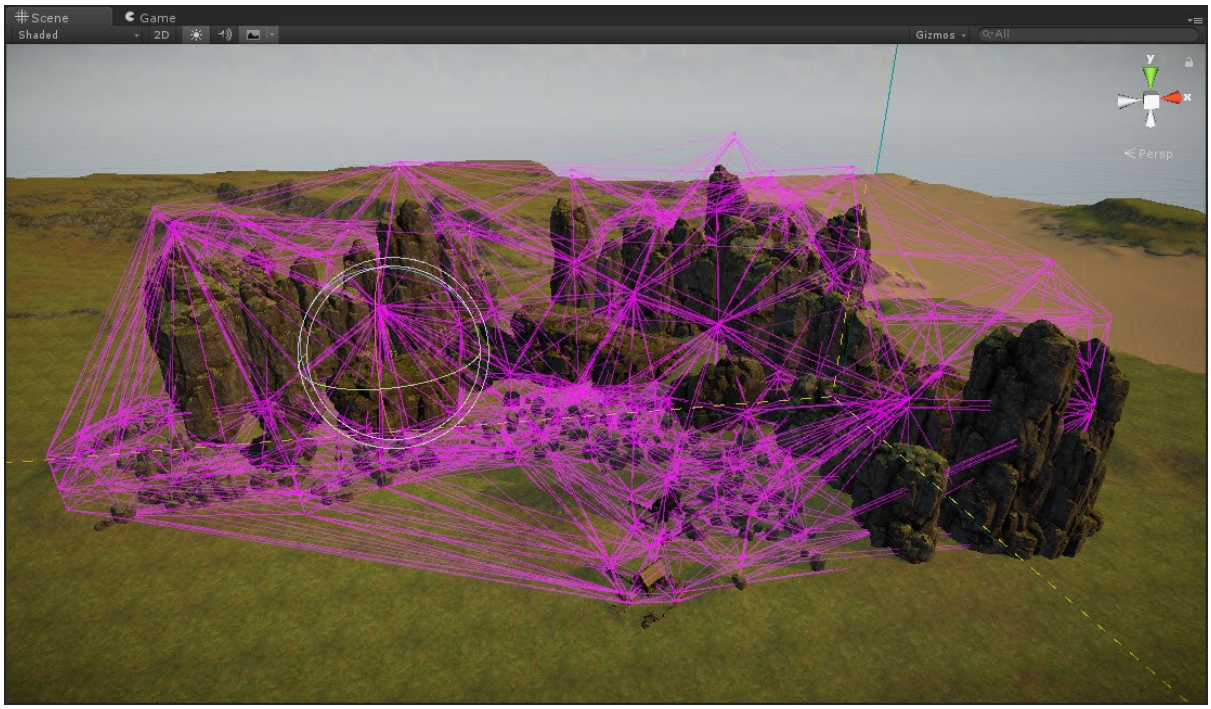
```
using GeNa.Core;
using UnityEditor;

// Custom Editor Attribute for GeNaGrowthDecorator
[CustomEditor(typeof(GeNaGrowthDecorator))]
Editor
public class GeNaGrowthDecoratorEditor : GeNaDecoratorEditor<GeNaGrowthDecorator> // Inherits from Custom Editor that Supports GeNa Pro
{
    Frequently called  0+1 usages
    protected override void RenderPanel(bool helpEnabled)
    {
        EditorGUI.BeginChangeCheck();
        {
            Decorator.speed = EditorGUILayout.Slider(label: "Speed", value: Decorator.speed, leftValue: 1f, rightValue: 10f);
            Decorator.startScale = EditorGUILayout.Vector3Field(label: "Start Scale", Decorator.startScale);
            Decorator.endScale = EditorGUILayout.Vector3Field(label: "End Scale", Decorator.endScale);
        }
        if (EditorGUI.EndChangeCheck())
        {
            EditorUtility.SetDirty(Decorator);
        }
    }
}
```

Step 6). Perform a Spawn!



Extras

Light Probe System



To get the best lighting out of the Unity Global Illumination (GI) system you should always allow your light to bake. The problem with this is that even a relatively simple scene can take hours or days.

The biggest culprit are small objects that have been set to Lightmap Static. This causes Unity to undergo a time and resource intensive baking process that is not only slow in the editor but can also cause performance issues at runtime.

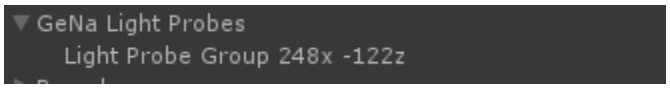
The solution to this is to use light probes. Light probes are light weight, fast, and minimise the need of lightmap baking. However, in addition to setting light probes up in your scene, the mesh renderers in the assets also need to be configured to use blend probes. When a prefab is spawned using the GeNa optimisation system it is also modified to use blend probes.

On top of this, many assets are not configured correctly to take advantage of Unity batching, and these can compound performance issues. When prefabs are spawned as optimised prefabs by GeNa they are also set up for batching.

The GeNa light-probe and optimization system addresses these issues by changing the way objects are spawned – it makes sure that the respective static flags are set optimally, that all mesh renderers have been configured to take advantage of light probes, and it automatically places light probes.

To enable the Light Probe system in your spawner go into Advanced Settings and ensure that Add Light Probes is selected. GeNa will use these settings to add light probes when you spawn Prefabs into your scene.

When GeNa creates light probes it will also create a GeNa Light Probes object in your scene and then parents each Light Probe Group under it. The naming standard appends the x & z location onto the light probe group so it's easy to locate and enable or disable individual probe groups.



The probes are added in stacks of 3 to best capture the light. One is ½ meter above the spawn point, one ½ meter above top of the prefab, and another 5 meters above that. The probe system enforces minimal distances between probe groups and probes so that you do not get too great a density of them, as this can be detrimental to performance.

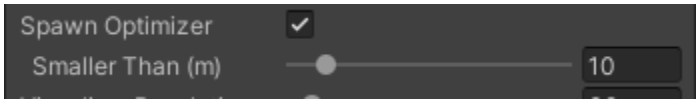
The **Min PG Dist** setting is the minimum distance between Light Probe Groups that are added to your scene.

The **Min Probe Dist** setting is the minimum distance between the Light Probes that are added to your scene. Adding too many light probes can be detrimental to performance, so do not make this setting too small.

Here is a great tutorial from Unity on Precomputed Realtime GI:
https://unity3d.com/learn/tutorials/topics/graphics/introduction-precomputed-realtime-gi?playlist=17102

Spawn Optimisation System

To enable the Spawn Optimisation system in your spawner go into Advanced Settings and ensure that Spawn Optimizer is selected. GeNa will use these configuration settings to automatically optimise prefabs as they are spawned into your scene.



Optimisation consists of ensuring that all meshes are set to use blend probes. You must also turn on the Light Probe System to take advantage of this unless you prefer to manually place your light probes.

The **Smaller Than (m)** setting is the size below which objects will be automatically optimised when they are spawned, however you can also influence and override this in your resource settings.

Here is the Unity documentation on batching: https://docs.unity3d.com/Manual/DrawCallBatching.html.

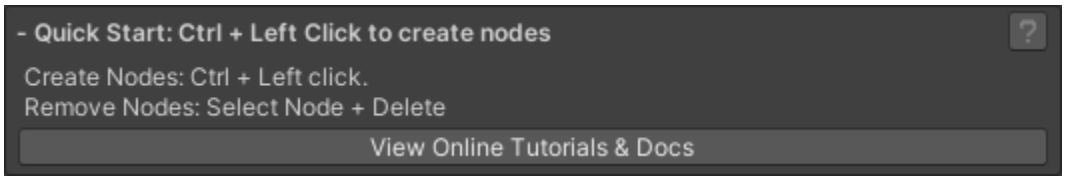
GeNa Spline

The spline system allows you to create splines and then apply operations into your scene using Spline extensions.

GeNa has 3 main types of spline. Generic splines, Road splines and River splines. Road and River splines have additional capability embedded into them to help in the creation of roads and rivers.

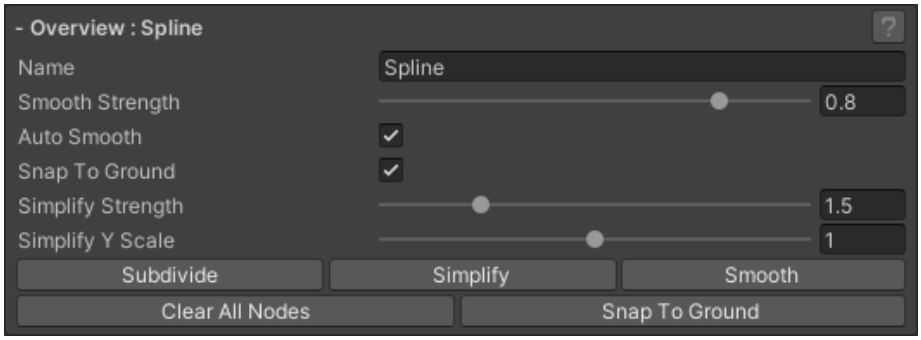
Interface

Quick Start Panel



The shortcuts panel shows the mouse and keyboard controls for the spline. It is there for convenience and can be toggled by clicking on the ***Question Mark*** at the top righthand side. For more information on the keyboard and mouse controls please see the Mouse & Keyboard controls section later in of the manual.

Overview Panel



Name: The name of the spline.

Smooth Strength: The smoothness strength of the curves between nodes.

Auto Smooth: Automatically smooths the curves between newly added nodes.

Snap to Ground: Automatically snaps nodes to ground when subdividing.

Simplify Strength: Strength of simplification (max distance between 0-5).

Simplify Y Scale: Amount of height difference simplification.

Subdivide: Subdivides the number of nodes.

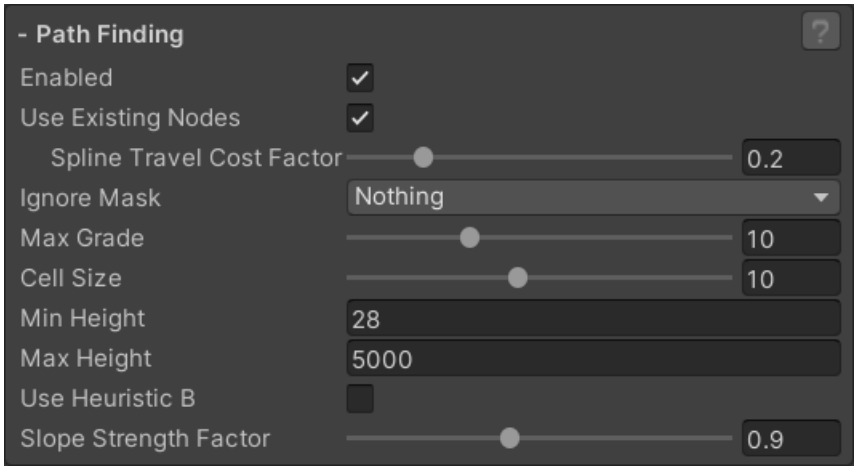
Simplify: Removes unnecessary nodes between line segments using approximation.

Smooth: Smooths the curves between nodes.

Clear All Nodes: Removes all nodes from spline.

Snap to Ground: Snaps all nodes to nearest ground height.

Pathfinding Panel



Enabled: Enable the Path Finding module for seeking parameterised paths between two nodes.

Use Existing Nodes: Enables the Path Finder to use existing nodes/curves to find a shorter path.

Spline Travel Cost Factor: The cost factor for traveling on a spline instead of bare land.

Ignore Mask: Layers to ignore during path finding.

Max Grade: The maximum uphill or downhill grade that is considered a valid move. (see also: Slope Strength Factor)

Cell Size: The rectangular size in meters that areas are considered for valid moves.

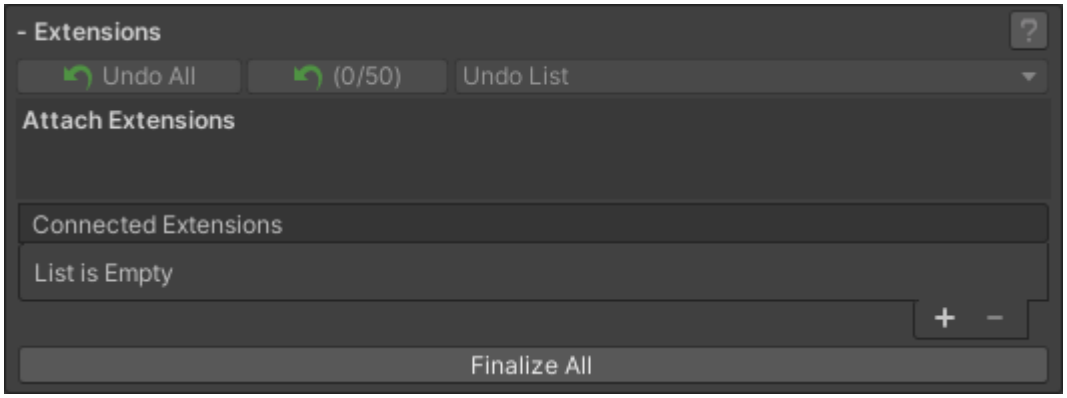
Min Height: Minimum height that the path will descend to.

Max Height: Maximum Height that the path will climb to.

Use Heuristic B: Use a different Heuristic for determine valid moves along the path.

Slope Strength Factor: How strong the local slope is considered (used with default Heuristic).

Extensions Panel



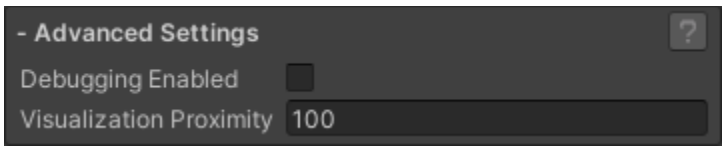
Undo: Undoes recorded actions of any Extensions.

Attach Extensions: Drag and Drop area for Extension objects.

Connected Extensions: Currently attached Extensions on the Spline.

Finalize All: Performs a Bake on all Extensions.

Advanced Settings Panel



Debugging Enabled: Shows any visualization for spline debugging

Visualization Proximity: Distance in world units where visualization is drawn.

Extensions

Spawner Extension

The Spawner Extension is used to perform GeNa Spawner operations along a spline.

Spawner: The connected Spawner object.

Flow Rate: The distance between spawn locations in world units.

Spawn Range: The maximum range of the spawner in meters. Prototypes will only be spawned in this range around the central spawn location.

NOTE: Gena will choose a random value between min and max. This allows successive iterations to spawn different numbers of instances which is a great way of introducing variability into your spawns.

Throw Distance: The distance that will be used by the various spawn algorithms to place newly spawned instances. How it is used will vary from algorithm to algorithm. Set this to zero when in Paint mode to have your spawned objects accurately follow your mouse.

Offset Position X: Position offset on the X axis for each spawn location.

Offset Rotation Y: Rotation offset around Y axis for each spawn location.

Modify Spawner: Allows Spawner editing in the Scene View.

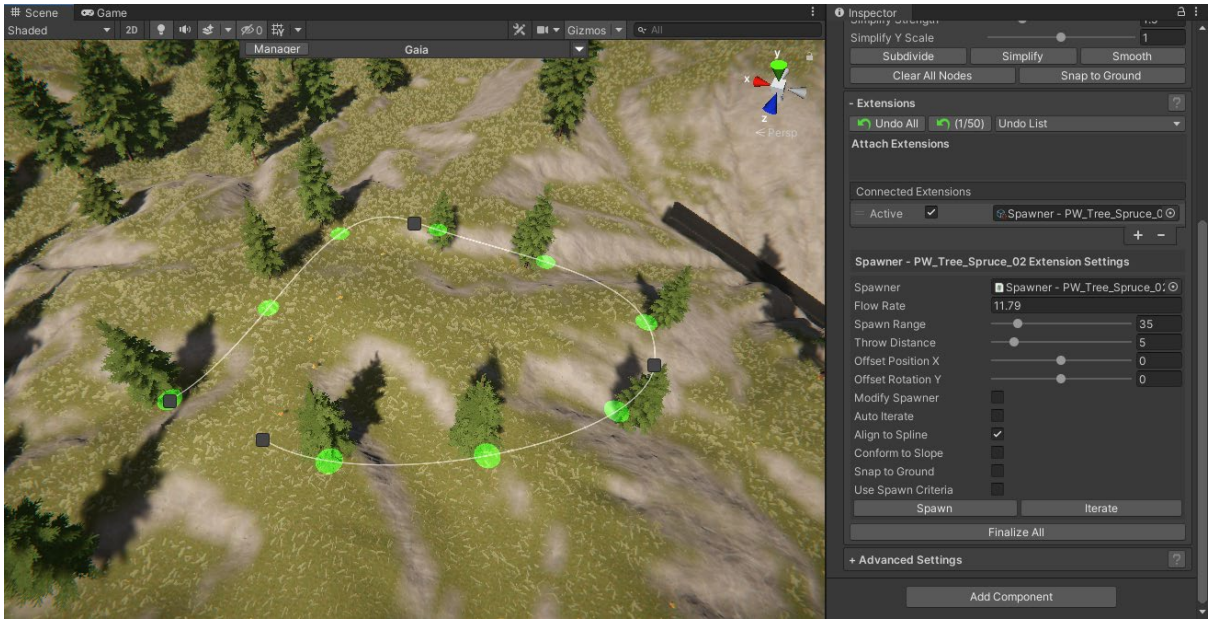
Auto Iterate: Performs Iterate procedure when adding new nodes.

Align to Spline: Aligns all spawned entities to spline (in rotation).

Conform to Slope: Conforms all spawned entities to slope along spline.

Snap to Ground: Snaps al spawned entities to nearest ground height.

Use Spawn Criteria: Updates visibility of objects based on Spawner's Criteria

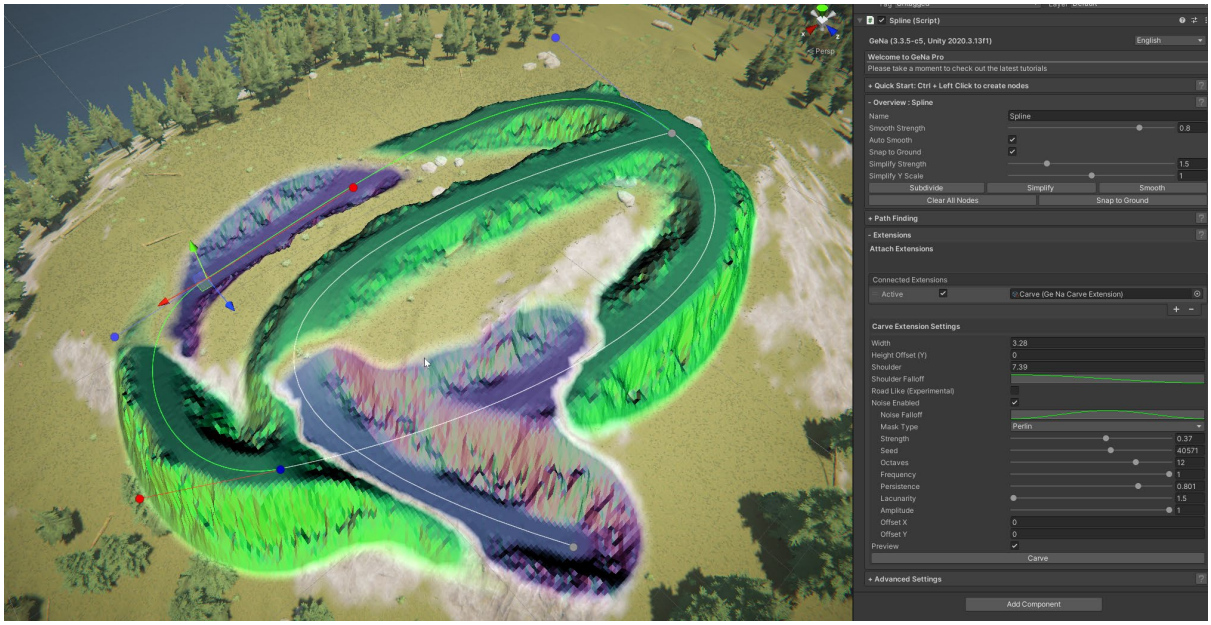


Spawn: Performs a Spawn operation on GeNa Spawner.

Iterate: Performs an Iterate operation on GeNa Spawner.

Carve Extension

The Carve Extension is used to create embankments in your Terrain.



Width: The width of the carve

Height Offset (Y): The offset of the carve operation in the Y axis

Shoulder: Shoulder width along the spline.

Shoulder Falloff: The falloff in strength of the shoulder along the spline.

Noise Enabled: Refer to the Spline Noise section.

Clear Colliders Extension

The Clear Colliders Extension is used to remove any Colliders along the Spline.

Width: The width of the removal along the Spline.

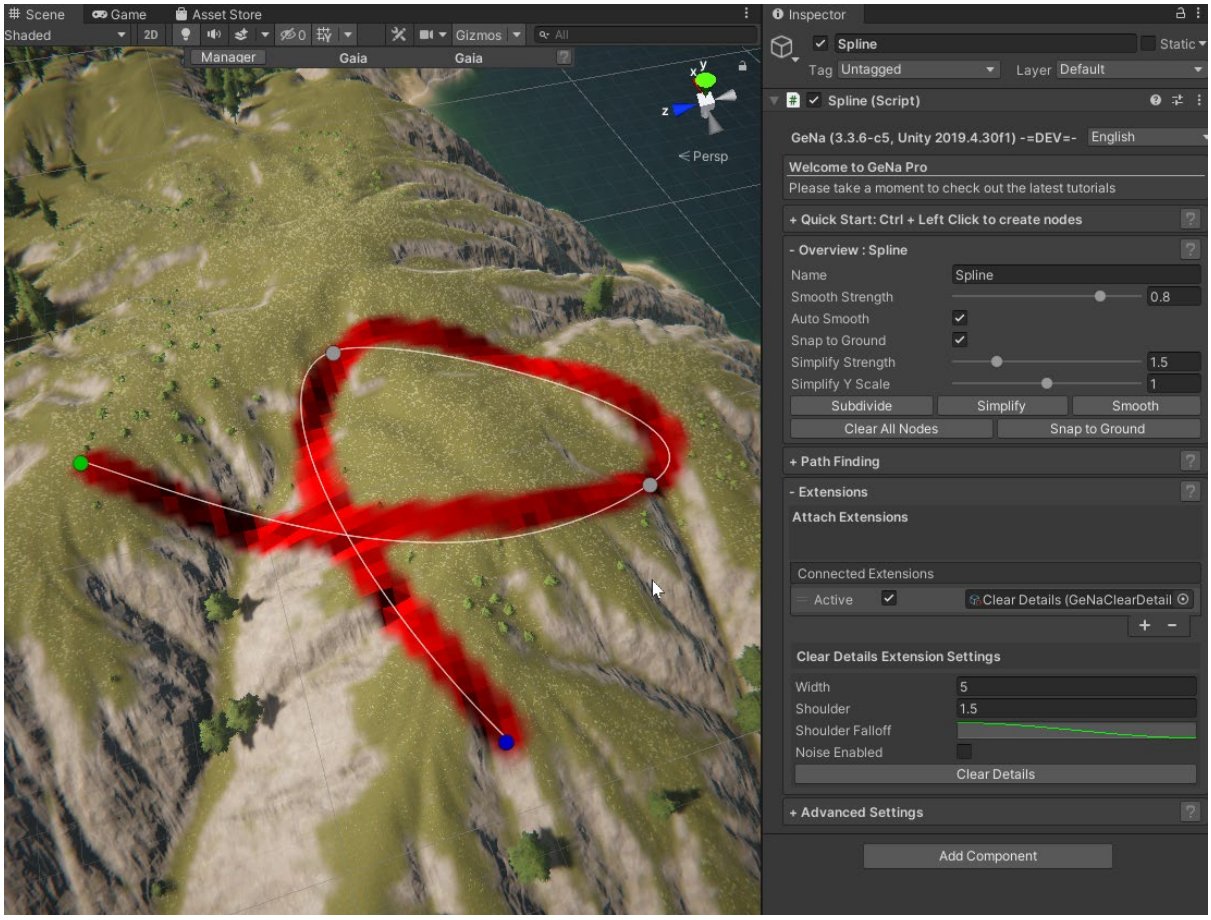
Layer Mask: Layer Mask for the collision detection.

Ignored Collisions: A list of colliders to ignore in the clear process.

Clear Colliders: Clears the colliders along the spline.

Clear Details Extension

The Clear Details Extension is used to remove any Terrain Details along the Spline.



Width: The width of the removal texture along the Spline.

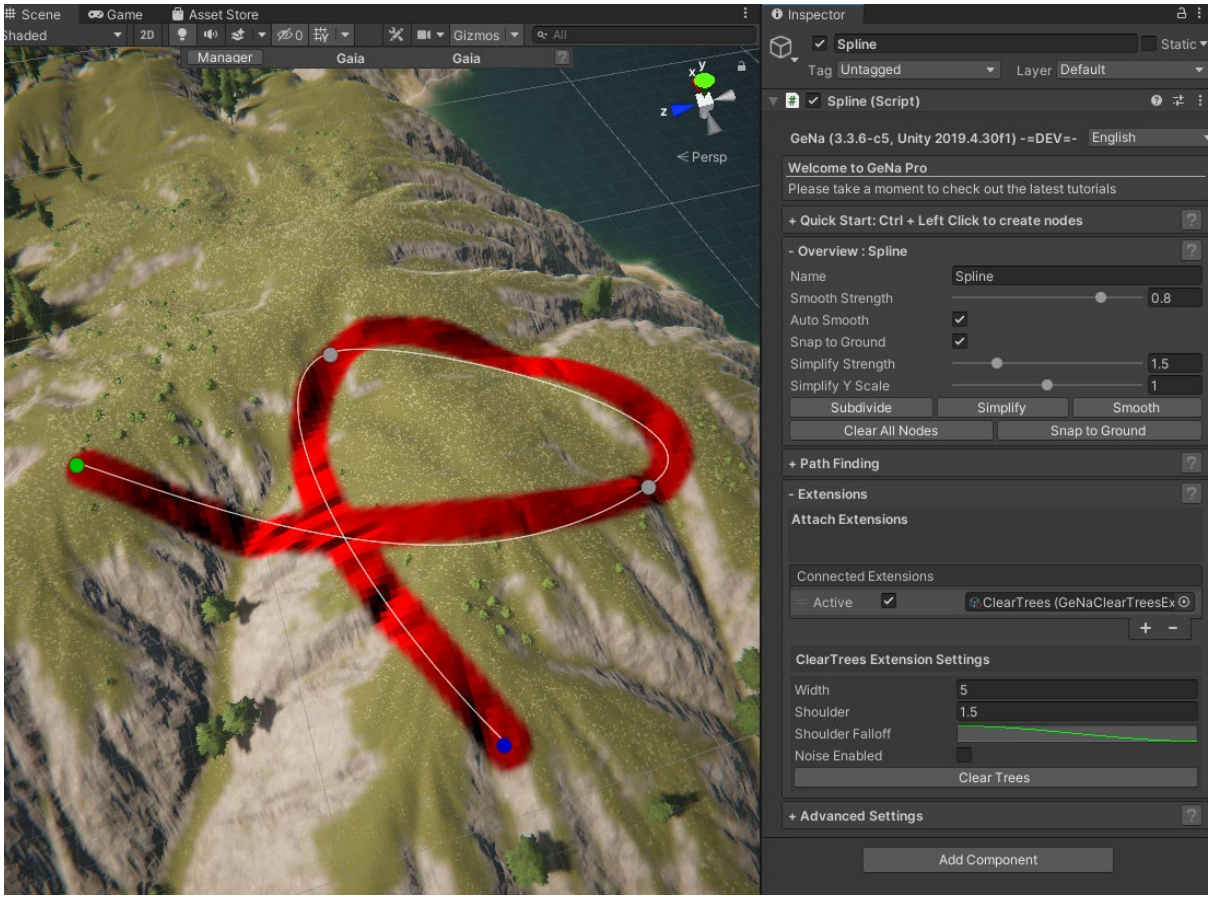
Shoulder: Shoulder width along the spline.

Shoulder Falloff: The falloff in strength of the shoulder along the spline.

Noise Enabled: Refer to the Spline Noise section.

Clear Trees Extension

The Clear Trees Extension is used to remove any Terrain Trees along the Spline.



Width: The width of the removal texture along the Spline.

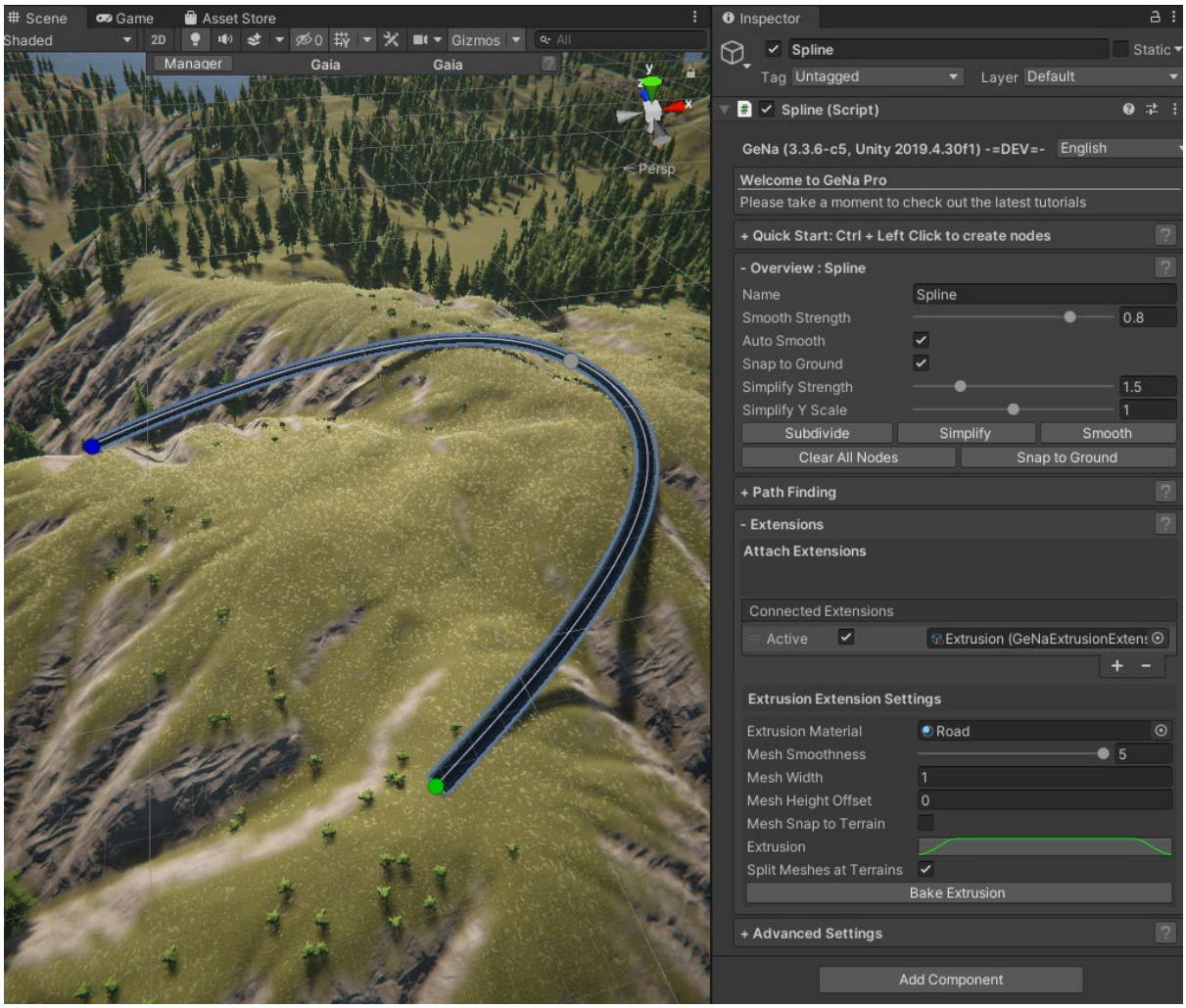
Shoulder: Shoulder width along the spline.

Shoulder Falloff: The falloff in strength of the shoulder along the spline.

Noise Enabled: Refer to the Spline Noise section.

Extrusion Extension

The Extrusion Extension is used to elongated meshes along the spline. Examples include Roads, Rivers, Bridges, etc.



Extrusion Material: The Material applied to the extrusion meshes.

Mesh Smoothness: Smoothness in vertices of the mesh along the spline.

Mesh Height Offset: Height offset of the mesh along the spline in the Y axis.

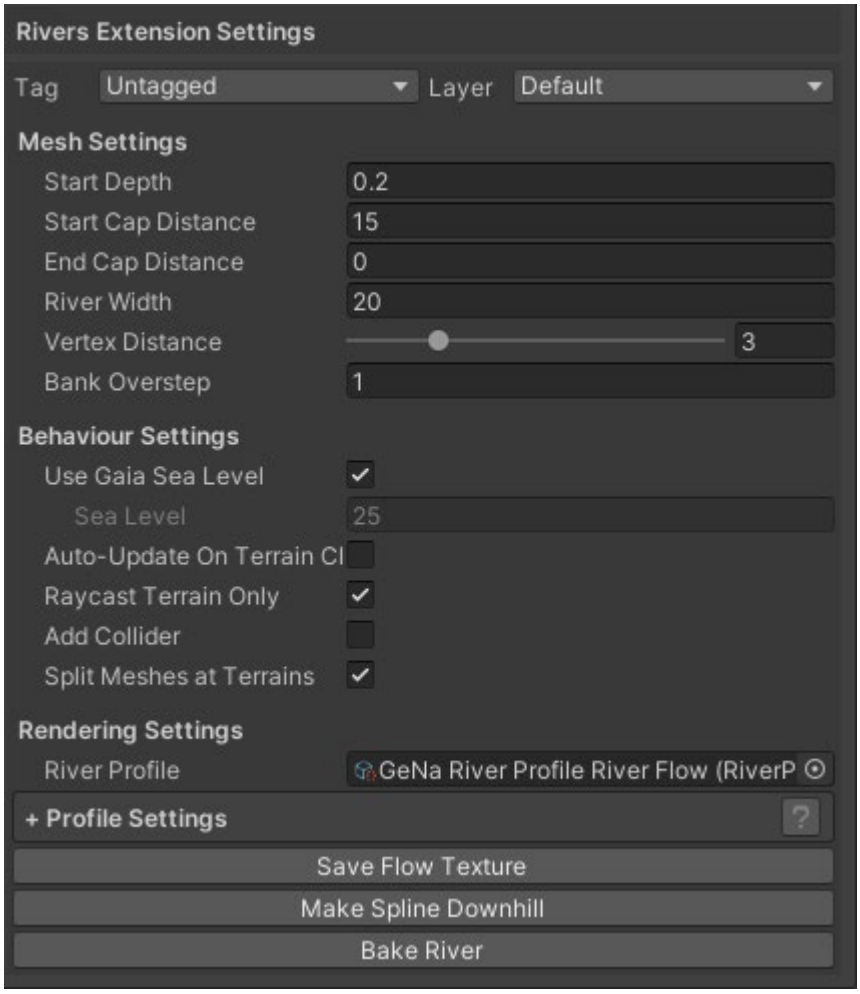
Mesh Snap to Terrain: Snap the edges of the extrusion along the spline to the Terrain.

Extrusion: Animation Curve representing the shape of the extrusion along the spline.

Split Meshes at Terrains: When you bake the extension, it will split the mesh up at the end of the terrain bounds. This will also parent the split baked mesh to the terrain it is on.

River Extension

The Rivers Extension is used to create high fidelity river flows along the spline.



Tag: Defines whether the river object should have a tag applied.

Layer: Defines whether the river object should have a layer applied.

Mesh Settings: Parameters that affect the way the River mesh is generated.

> **Start Depth:** Sets how high the river flow starts at. The higher the number the higher the river mesh will be from the ground level.
>
> **Start Cap Distance:** Sets the start nodes cap distance. If the cap is sticking out of the terrain you can control how far this cap goes with this paramater.
>
> **End Cap Distance:** Adds a level end-cap to the lower part of rivers, when > 0.
>
> **River Width:** Sets how wide the river can be. Note that if the river hits a embankment it will no longer extend it's width.
>
> **Vertex Distance:** How far apart (along the river spline) it will create a quad (and then triangulate it).

Bank Overstep: How far into the terrain it will pierce. It is an advanced setting. Normally 1 is a good value (1m). But it can be lowered (or raised) if it sticks out in some situations where a thin raised bank shows the river mesh on the opposite side of the bank.

Worldspace Tiling: River mesh creation takes the worldspace width into consideration if enabled.

Tile X: The distance across the width of the river that the river texture/normal repeats.

Behaviour Settings: Parameters that affect the way the river mesh behaves / interacts with its environment.

Use Gaia Sea Level: If enabled the river will use the gaia sea level. Otherwise it will use the Sea level value you setup.

Sea Level: Sets the water level for the river where it will start to blend into the ocean.

Auto-Update On Terrain Change: The River Mesh updates immiately after each terrain change, if enabled.

Add Collider: Adds a mesh collider to the mesh created by this extension.

Split Meshes At Terrains: When you bake the extension it will split the mesh up at the end of the terrain bounds. This will also parent the split baked mesh to the terrain it is on.

Rendering Settings: Parameters that affect the appearance of the river mesh using River Profiles.

River Profile: This profile defines the shader and material settings for this river.

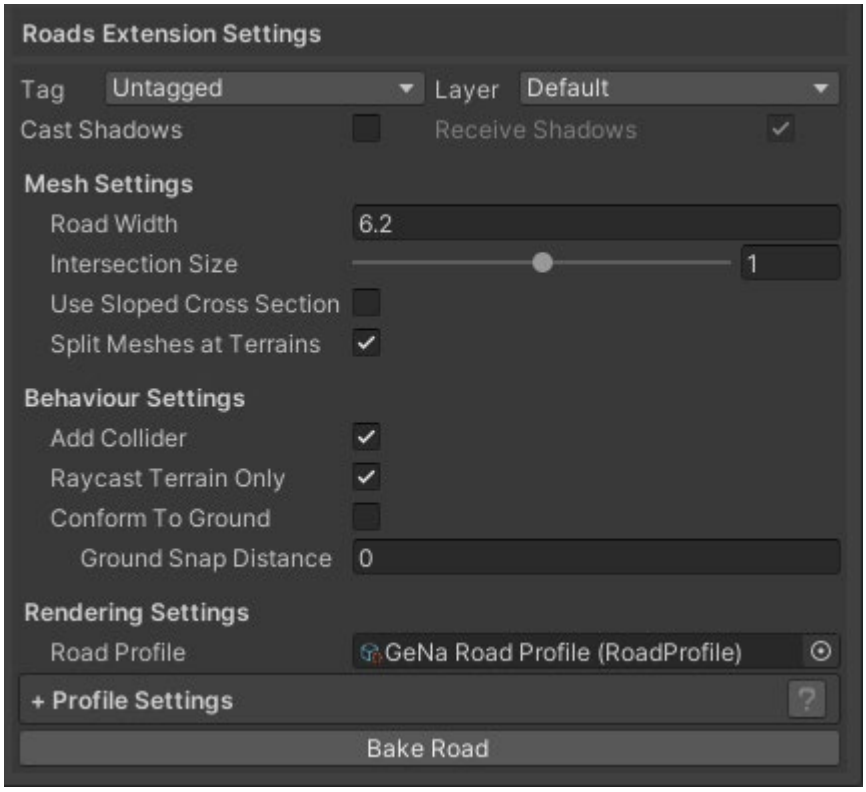
Profile Settings: See the River Profile section.

Make Spline Downhill: Forces the spline nodes on this river to always flow downhill.

Bake River: This will bake this river disabling GeNa spline but saving the generated mesh in the scene. NOTE: Gaia Terrains as Scenes requires Baking before play/build.

Road Extension

The roads extension is used to create high quality road meshes along the spline.



Tag: Defines how the tag should be setup on this object.

Layer: Defines how the layer should be setup on this object.

Cast Shadows: Should shadows be cast by these meshes?

Receive Shadows: Should these meshes receive shadows? Note: This will be greyed out if your Rendering Path is not set to 'Forward'.

Mesh Settings: Parameters that affect the way the Road mesh is generated.

> **Road Width:** How wide the road mesh is.
>
> **Intersection Size:** The size of how big intersections can be.
>
> **Use Sloped Cross Section:** Use a sloped cross section profile instead of a hard edged profile.
>
> **Split Meshes at Terrains:** When it bakes the road, it will split the mesh up where the end of the terrain. This will also parent the split baked mesh to the terrain it is on.

Behaviour Settings: Parameters that affect the way the road mesh behaves / interacts with its environment.

Add Collider: Adds a mesh collider to this spline.

Raycast Terrain Only: If enabled, the mesh will always stick to the terrain and conform the mesh shape to the shape of the terrain.

Conform to Ground: If enabled, the road will always conform to the ground.

Ground Snap Distance: Distance above the terrain before snapping activates.

Rendering Settings: Parameters that affect the appearance of the river mesh using Road Profiles.

Road Profile: The profile that settings used to set the road materials up.

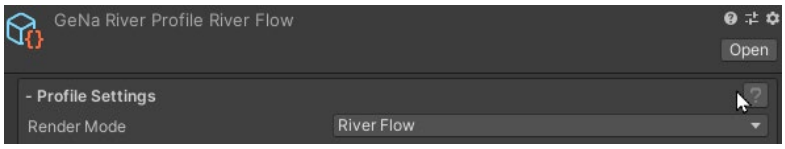
Profile Settings: See the Road Profile section.

Split Meshes at Terrains: Will split road meshes at the edge of terrains when baking.

Bake Road: Bake the road mesh into the scene and destroy the road spline.

Profiles

River Profile

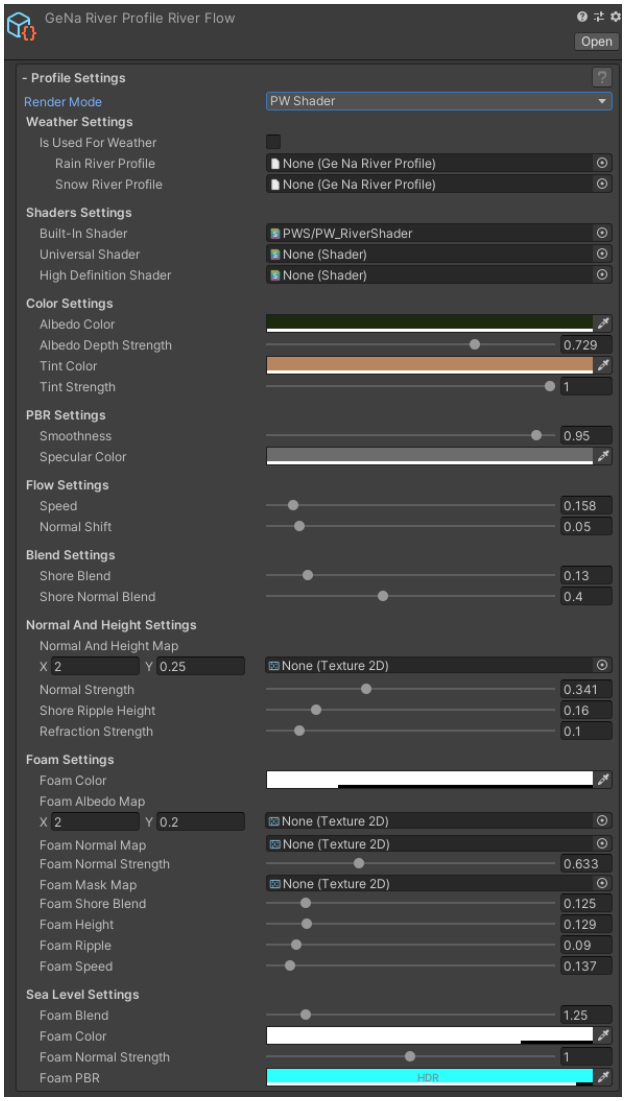


The river profile describes the appearance of the river mesh along the spline. It has a number of modes with different sets of parameters.

Render Mode: Sets what mode the profile will render.

> **PW Shader Mode** - Builds a new material instance and apply our profile settings.
>
> **Material Mode** – Creates a new material instances based on the material supplied shader then it will copy all the material properties over.
>
> **River Flow Mode** – Creates a new material based on the 'River Flow Material' with a whole new set of properties.

PW Shader Mode



Builds a new material instance and apply our profile settings.

Weather Settings: The settings that are applied when using a weather system.

Is Used For Weather: If enabled the this profile is treated like a weather profile for the river. When it rain or snows the settings will be lerped to these values on the river material.

Rain River Profile: A profile that contains the rain version of the river.

Snow River Profile: A profile that contains the snow version of the river.

Shader Settings: The default shaders to use for each pipeline.

Built-In Shader: Sets the river built-in shader.

Universal Shader: Sets the river universal shader.

High Definition Shader: Sets the river high definition shader.

Color Settings: Controls the color of the river.

Albedo Color: Base color of the river.

Albedo Depth Strength: Depth strength of the river color.

Tint Color: Tint color of the river.

Tint Strength: Tint strength of the river.

PBR Settings: The PBR color settings for the water.

Smoothness: The smoothness of the river.

Specular Color: The highlight color of the water.

Blend Settings: Settings for edge blending of the river.

Shore Blend: The amount of transparency at the edge of the river.

Shore Normal Blend: The intensity of the normals at the edge of the river.

Normal and Height Settings: Settings to adjust the normal and height of the river.

Normal and Height Map: The normal texture to apply to the river.

Normal is contained in the RGB channels, Heightmap is contained in the Alpha channel.

Shore Ripple Height: Visibility of the ripples at the edges of the river.

Refraction Strength: The strength of refraction.

Foam Settings: Controls the appearance of foam along the river.

Foam Color: The color of the foam.

Foam Albedo Map: The diffuse texture of the foam.

Foam Normal Map: The normal texture of the foam.

Foam Normal Strength: The intensity of the normal for the foam.

Foam Mask Map: The mask to apply to the foam.

Each channel stores a different modification to the mask.

Red Channel - Specular

Green Channel - Occlusion

Blue Channel - Heightmap

Alpha Channel - Smoothness

Foam Shore Blend: The visibility of the foam at the edge of the river.

Foam Height: The visible height of the foam.

Foam Ripple: The intensity of the foam ripple.

Foam Speed: The speed of the foam.

Sea Level Settings: Controls the foam interaction with the Sea Level.

Foam Blend: The visibility of the foam based on the Sea Level.

Foam Color: The color of the foam.

Foam Normal Strength: The normal intensity of the foam.

Foam PBR: The PBR color of the foam when it reaches the Sea Level.

Each channel stores a different modification to the PBR color.

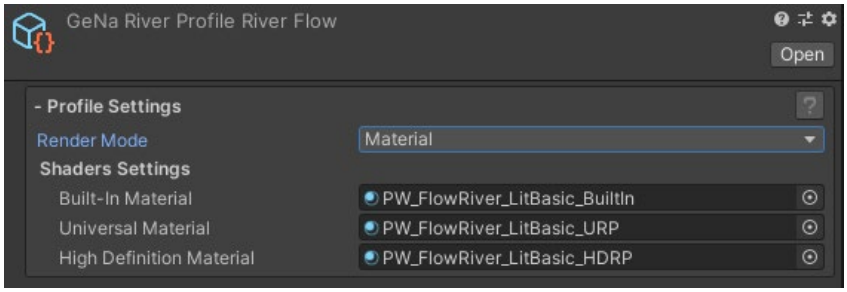
Red Channel - Specular

Green Channel - Occlusion

Blue Channel - Heightmap

Alpha Channel - Smoothness

Material Mode



Creates a new material instances based on the material supplied shader then it will copy all the material properties over.

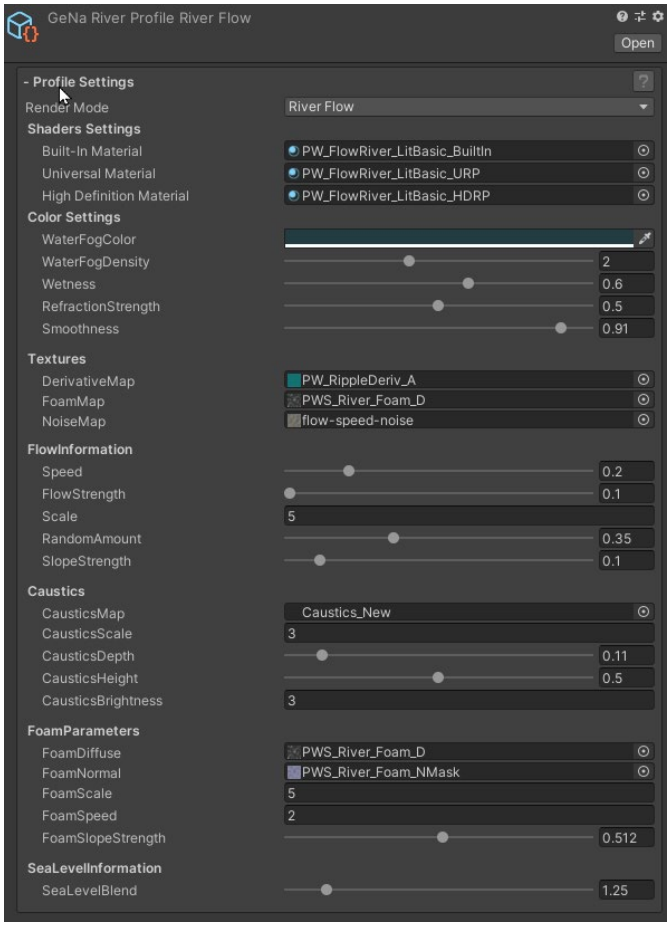
Shaders Settings: The default shaders to use for each pipeline.

Built-In Material: Sets the river built-in shader.

Universal Material: Sets the river universal shader.

High Definition Material: Sets the river high definition shader.

River Flow Mode



Creates a new material based on the 'River Flow Material' with a whole new set of properties.

Shaders Settings: The default shaders to use for each pipeline.

Build-In Material: Sets the river built-in shader.

Universal Material: Sets the river universal shader.

High Definition Material: Sets the river high definition shader.

Color Settings: Controls the color of the river.

Water Fog Color: Color of the dense water.

Water Fog Density: The density of the water.

Wetness: Controls how wet objects intersecting the water surface become.

Refraction Strength: Controls the strength of the refraction under the water.

Smoothness: The smoothness of the water.

Textures: A list of textures used by the material.

Derivative Map: The map used for the water normals. Normal derivatives XY and are stored in the Alpha and Green channel respectively.

Foam Map: The diffuse texture to use for the foam.

Noise Map: A noise map used for pseudo random calculations.

Flow Information: Parameters for controlling the flow of the river.

Speed: How fast the river travels.

Flow Strength: Increases the distributed strength of the river normals.

Scale: The scale of the river details.

Random Amount: Controls the spread on how much each angle gets randomised by. A higher value will produce widly varying angles between each direction.

Slope Strength: Controls how fast the river flows based on the slope. A lower value will produce a faster speed on sloped areas.

Caustics Settings: Adjustments to the appearence of the caustics.

Caustics Map: The texture to use for displaying underwater caustics.

Caustics Scale: The scale of the caustics map.

Caustics Depth: The depth at which the caustics get mapped to. A lower value will produce more extreme distortion.

Caustics Height: Controls how distorted the caustics become based on the height of the water surface.

Caustics Brightness: Controls how bright to display the caustics.

Foam Settings: Controls the appearance of foam along the river.

Foam Diffuse: The diffuse texture to use for the foam surface.

Foam Normal: The normal texture to use for the foam surface.

Foam Scale: The scale of the foam texture.

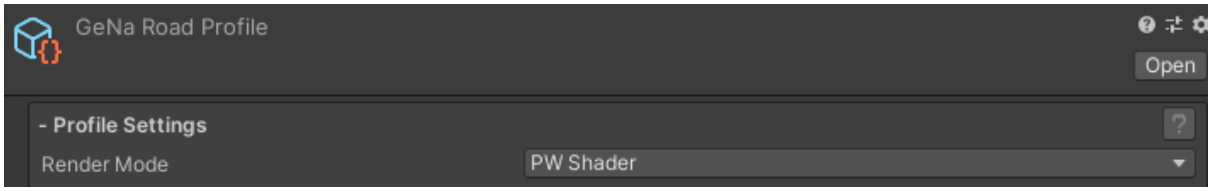
Foam Speed: The speed of the foam.

Foam Slope Strength: Controls the amount of foam that appears at slopes of the river.

Sea Level Information: Controls the foam interaction with the Sea Level.

Sea Level Blend: The visibility of the foam based on the Sea Level.

Road Profile

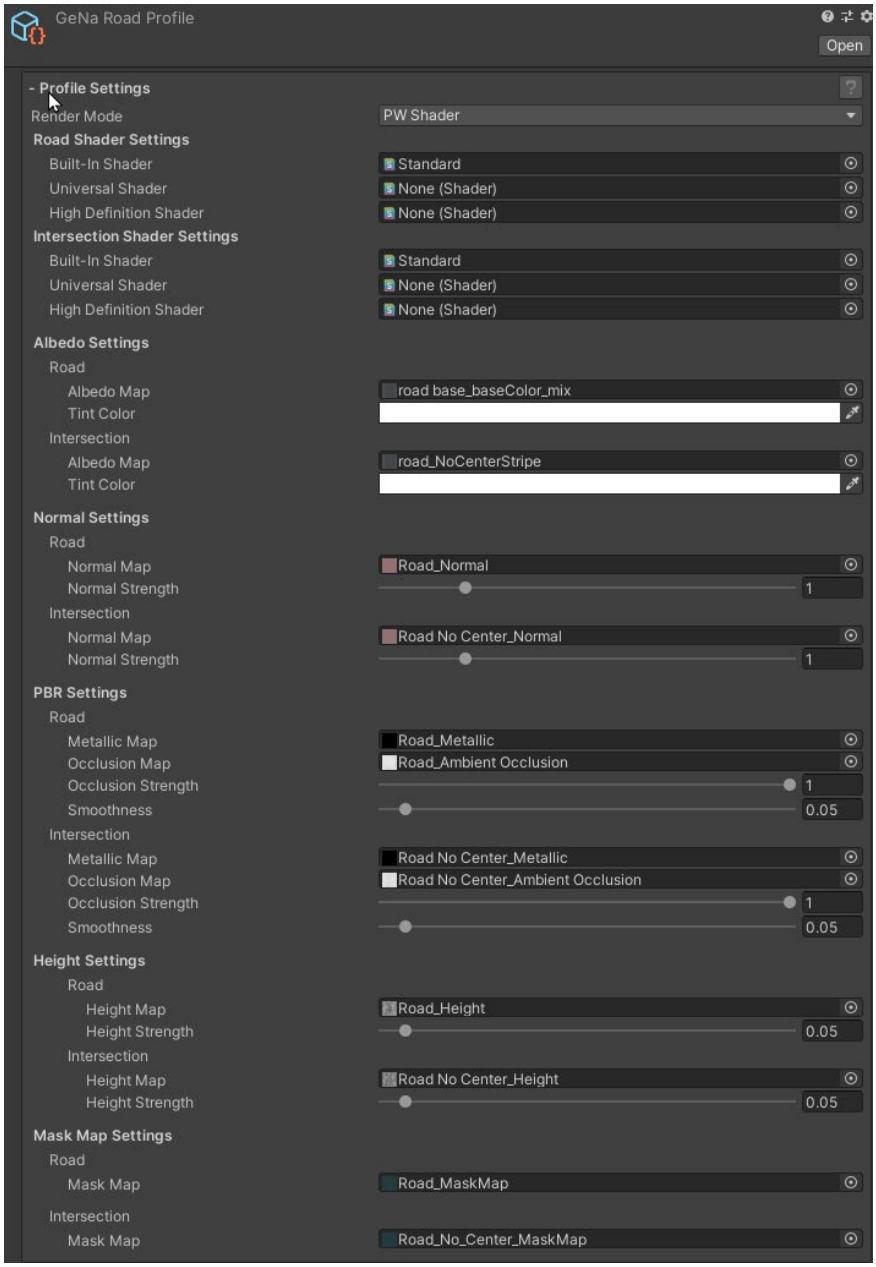


The road profile describes the appearance of the road mesh along the spline. It has a number of modes with different sets of parameters.

Render Mode: Sets what mode the profile will render.

> **PW Shader Mode** - Builds a new material instance and apply our profile settings.
>
> **Material Mode** – Creates a new material instances based on the material supplied shader then it will copy all the material properties over.
>
> ~~**River Flow Mode**~~ – Disabled for Road Profiles.

PW Shader Mode



Builds a new material instance and apply our profile settings.

Road Shader Settings: Default road shaders to use for each pipeline.

Built-In Shader: Sets the road built-in shader.

Universal Shader: Sets the road universal shader.

High Definition Shader: Sets the road high definition shader.

Intersection Shader Settings: Default intersection shaders to use for each pipeline.

Built-In Shader: Sets the intersection built-in shader.

Universal Shader: Sets the intersection universal shader.

High Definition Shader: Sets the intersection high definition shader.

Albedo Settings: Parameters to affect the appearance of the albedo color.

Road & Intersection: The following settings are for Roads and Intersections.

Albedo Map: Sets the albedo map texture.

Tint Color: Sets the tint color.

Normal Settings: Parameters to affect the appearance of the road normals.

Road & Intersection: The following settings are for Roads and Intersections.

Normal Map: Sets the normal map texture.

Normal Strength: Sets the strength of the normal map.

PBR Settings: Parameters to affect the appearance of the PBR outputs.

Road & Intersection: The following settings are for Roads and Intersections.

Metallic Map: Sets the metallic map (RGB: Metallic, A: Smoothness)

Occlusion Map: Sets the occlusion map texture.

Occlusion Strength: Sets the strength of the occlusion map.

Smoothness: Sets the smoothness scale factor.

Height Settings: Parameters to affect the appearance of the height output.

Road & Intersection: The following settings are for Roads and Intersections.

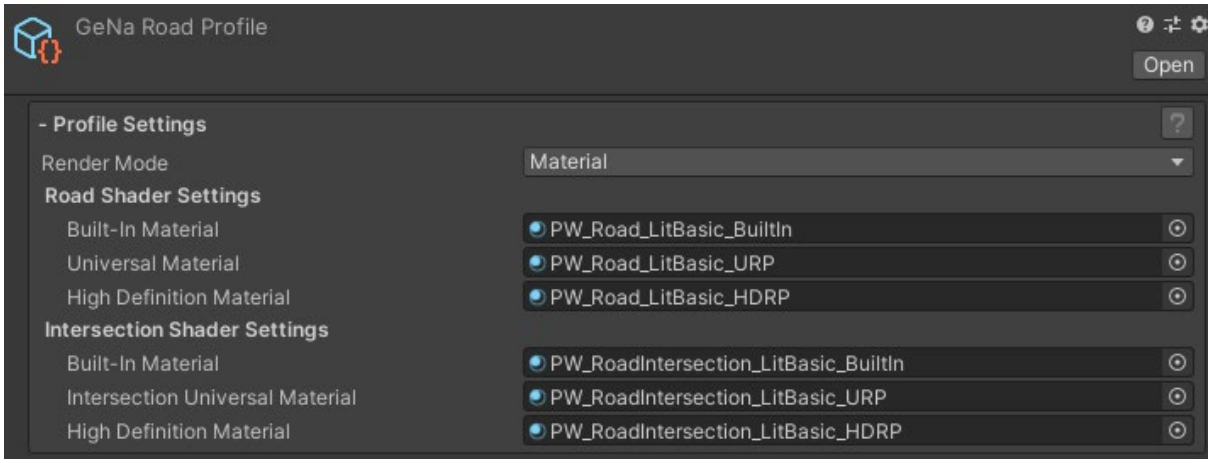
Height Map: Sets the height map texture.

Height Strength: Sets the strength of the height map.

Mask Map Settings: Parameters to affect the appearance of the PBR outputs.

Road & Intersection: The following settings are for Roads and Intersections.

Material Mode



Creates a new material instances based on the material supplied shader then it will copy all the material properties over.

Road Shader Settings: Default road shaders to use for each pipeline.

Built-In Shader: Sets the road built-in shader.

Universal Shader: Sets the road universal shader.

High Definition Shader: Sets the road high definition shader.

Intersection Shader Settings: Default intersection shaders to use for each pipeline.

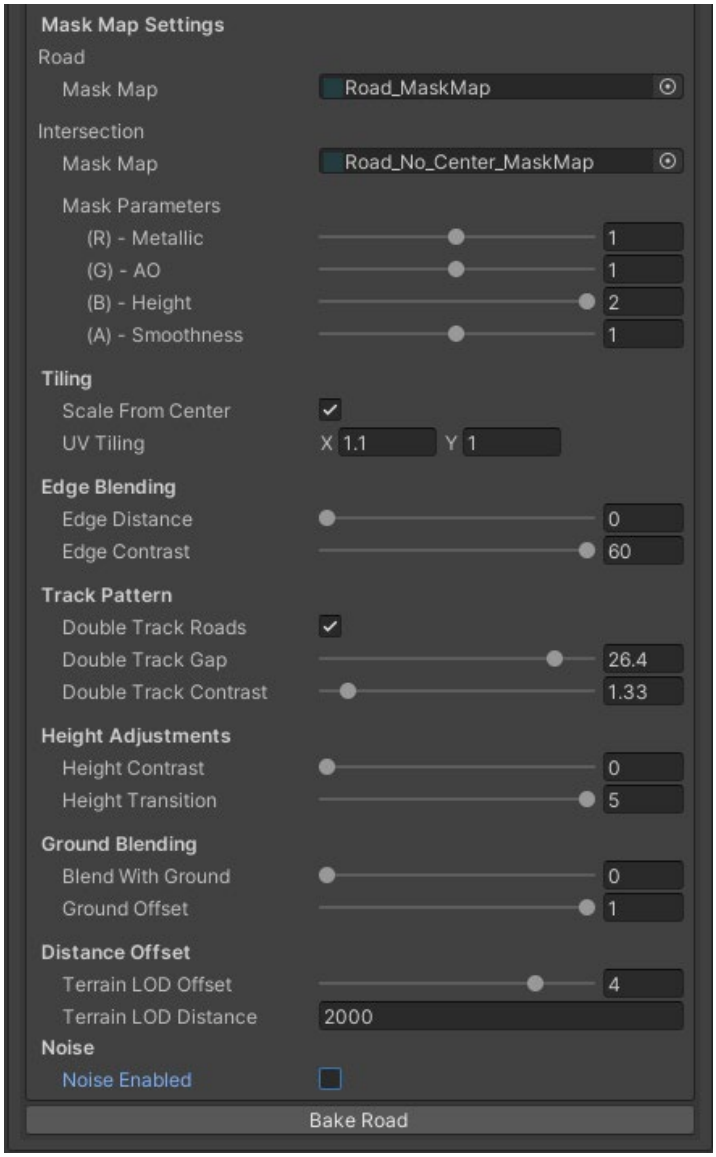
Built-In Shader: Sets the intersection built-in shader.

Universal Shader: Sets the intersection universal shader.

High Definition Shader: Sets the intersection high definition shader.

HDRP / URP

There are alternate settings for the Road Profile when you're using HDRP / URP.



Mask Map Settings: Parameters to affect the appearance of the PBR outputs.

Road & Intersection: The following settings are for Roads and Intersections.

Mask Map: Sets the mask map.

Mask Parameters: Allows you to set the Metallic, Ambient Occlusion, Detail Mask (Height) and Smoothness.

Tiling: Parameters used for adjusting the tiling of the road textures.

Scale From Center: True - Scales the UVs from the center of the mesh. Tiling will shrink inwards towards the center.

False – Scales the UVs from the bottom left of the UV coordinates.

UV Tiling: Separate tiling in both the X and Y coordinate of the UVs.

Edge Blending: Settings for blending the road into the terrain.

Edge Distance: The size of the edge blend from the middle of the road.

Edge Contrast: The falloff size of the edge blend.

Track Pattern: Settings for changing the look of the road.

Double Track Roads: True – Turns the road into two separate tracks. False – Allows middle split to be adjusted.

Double Track Gap: Describes the size of the two lanes.

Double Track Contrast: Describes the falloff of the two lane sizes.

Height Adjustments: Settings for adjusting the height blending.

Height Contrast: Controls the contrast in the height map used for blending the fading below its surface.

Height Transition: Controls the transition phase between in the height map.

Ground Blending: Parameters used for blending the road into the ground.

Blend with Ground: The amount to blend the road into the ground.

0 – No blend with the ground.

1 – Full blend with the ground based off the heightmap's adjusted settings.

Ground Offset: The amount the road is offset from the ground. The effect is applied on flat areas of the terrain.

Distance offset: Settings for the terrain offset of the road at a distance.

Terrain LOD Offset: Controls how high the road will be pushed upwards depending ` on the distance away from the terrain.

Terrain LOD Distance: Controls the distance away from the terrain at which the road will be pushed upwards.

Noise Enabled: Refer to the Spline Noise section.

Usage

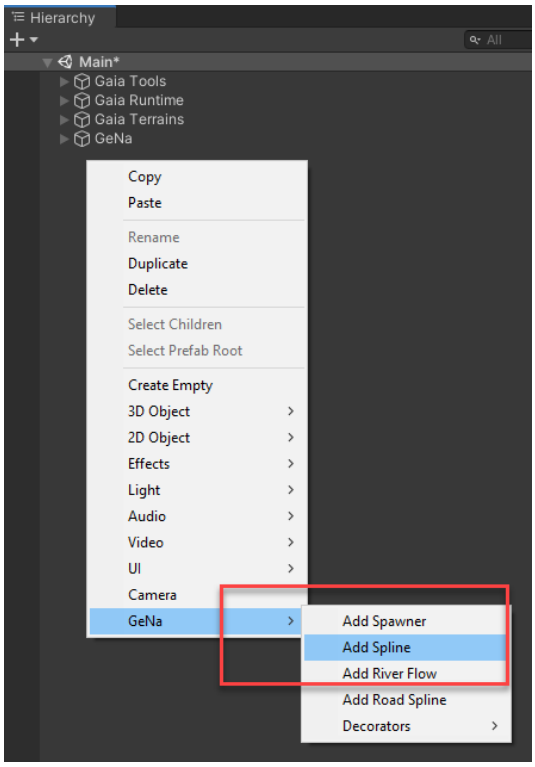
This section details some of the common usage scenarios for Splines and makes some suggestions on how to get better results.

Create a GeNa Spline

This work flow will act also work the same way for river and road splines.

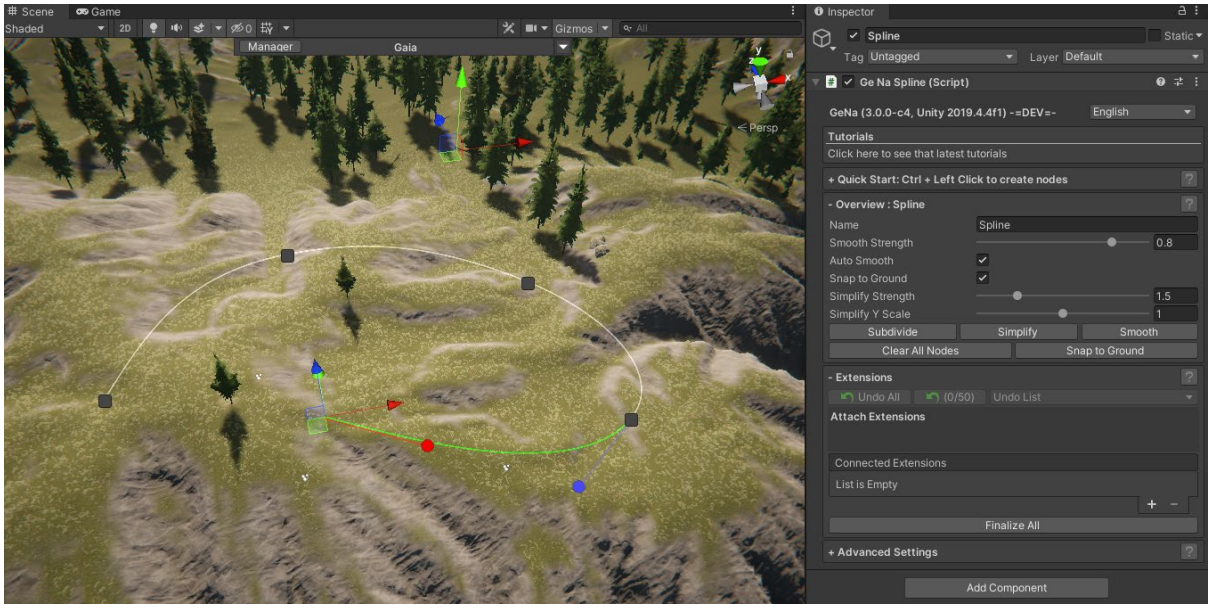
Step 1). **Create a GeNa Spline GameObject**

Right-Click on your **Hierarchy** and Select **GeNa > Add Spline**



Step 2). **Create Spline Nodes**

To add nodes **CTRL** and **Left-Mouse Click** anywhere on Terrain or GameObject in the Scene to add nodes.



Step 3). **Remove Spline Nodes**

Select a grey spline node by clicking on it. Hit the DEL key to delete it. This will split the spline.

Step 4). **Join Spline Nodes (Loop Splines)**

To join a spline node to another node select the first grey node by left clicking on it, and then Ctrl Left Click to join it to the node at the other end.

Step 5). **Subdivide Spline Nodes**

To add more detail to your spline you can subdivide it by clicking the subdivide button.

Step 6). **Smooth Spline Nodes**

To smooth splines click on the smooth splines button.

Step 7). **Simplify Spline Nodes**

To smooth splines click on the Simplify splines button. This will remove spline nodes while attempting to keep the same basic shape of the spline.

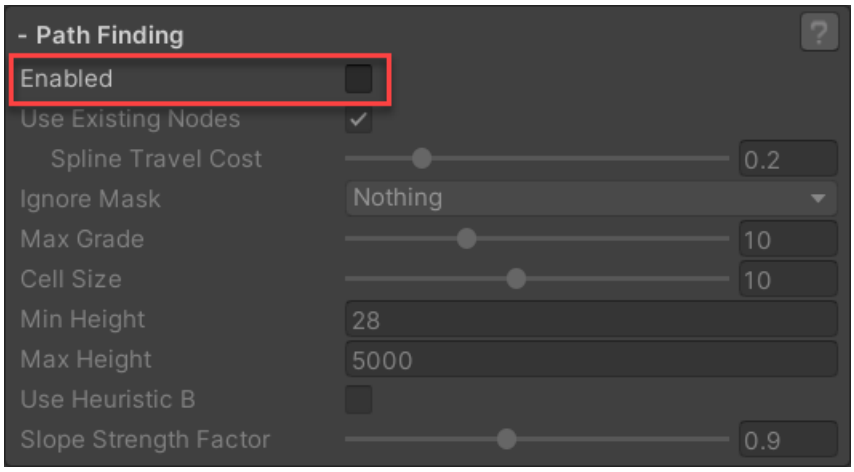
Step 8). **Change Spline Tangent Angles**

To change the angle that a spline joins another spline, click on the Grey Spline Node, and then select either the red or the blue tangent's and move them around.

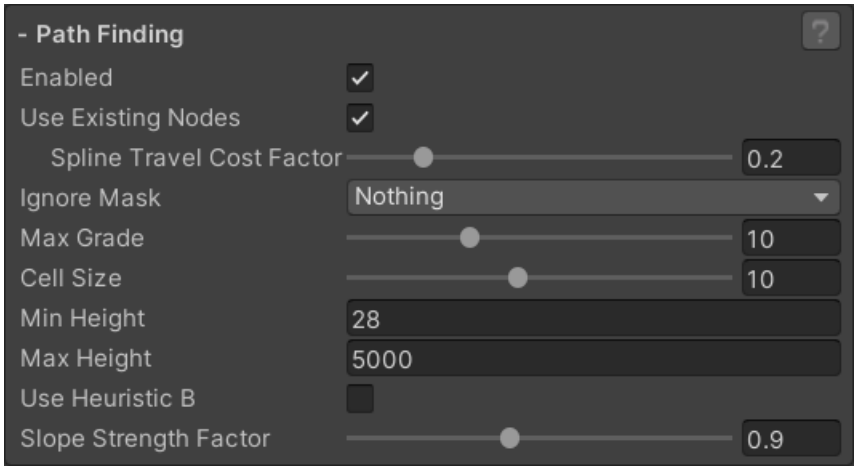
Using Path Finding

Step 1). Enable Path Finding

Before you can use Path Finding in your project, you will need to enable it.



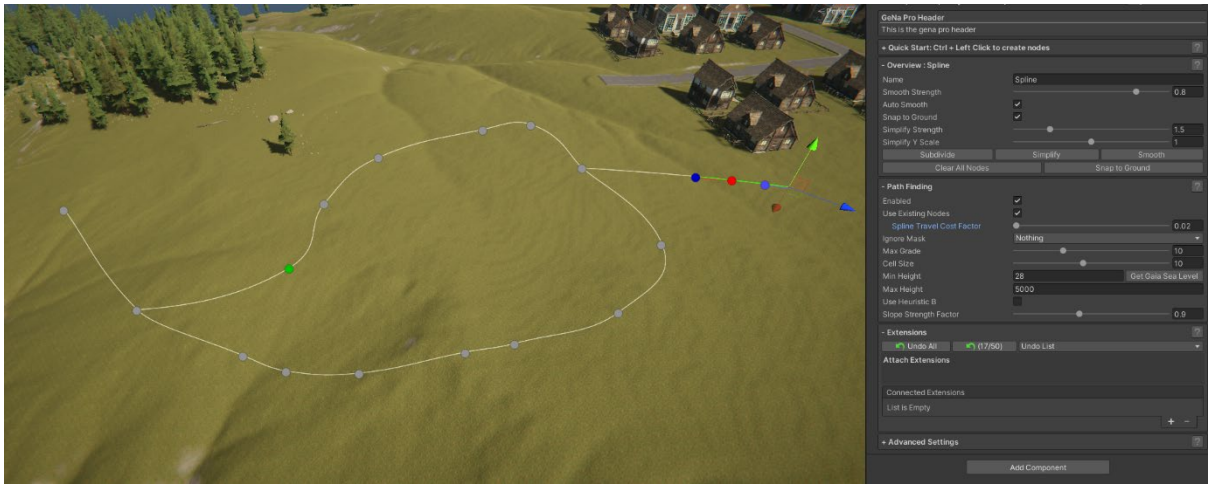
Once enabled you can modify settings of the Path finding system.



Step 2). Create Nodes

Create nodes the usual way by **CTRL + Left** Clicking onto your Terrain and GeNa will perform the Path Finding algorithm with the configuration you've set up above.

You can also select previously created nodes and GeNa will use the 'Travel Cost Factor' to re-use nodes in the shortest path algorithm like so.

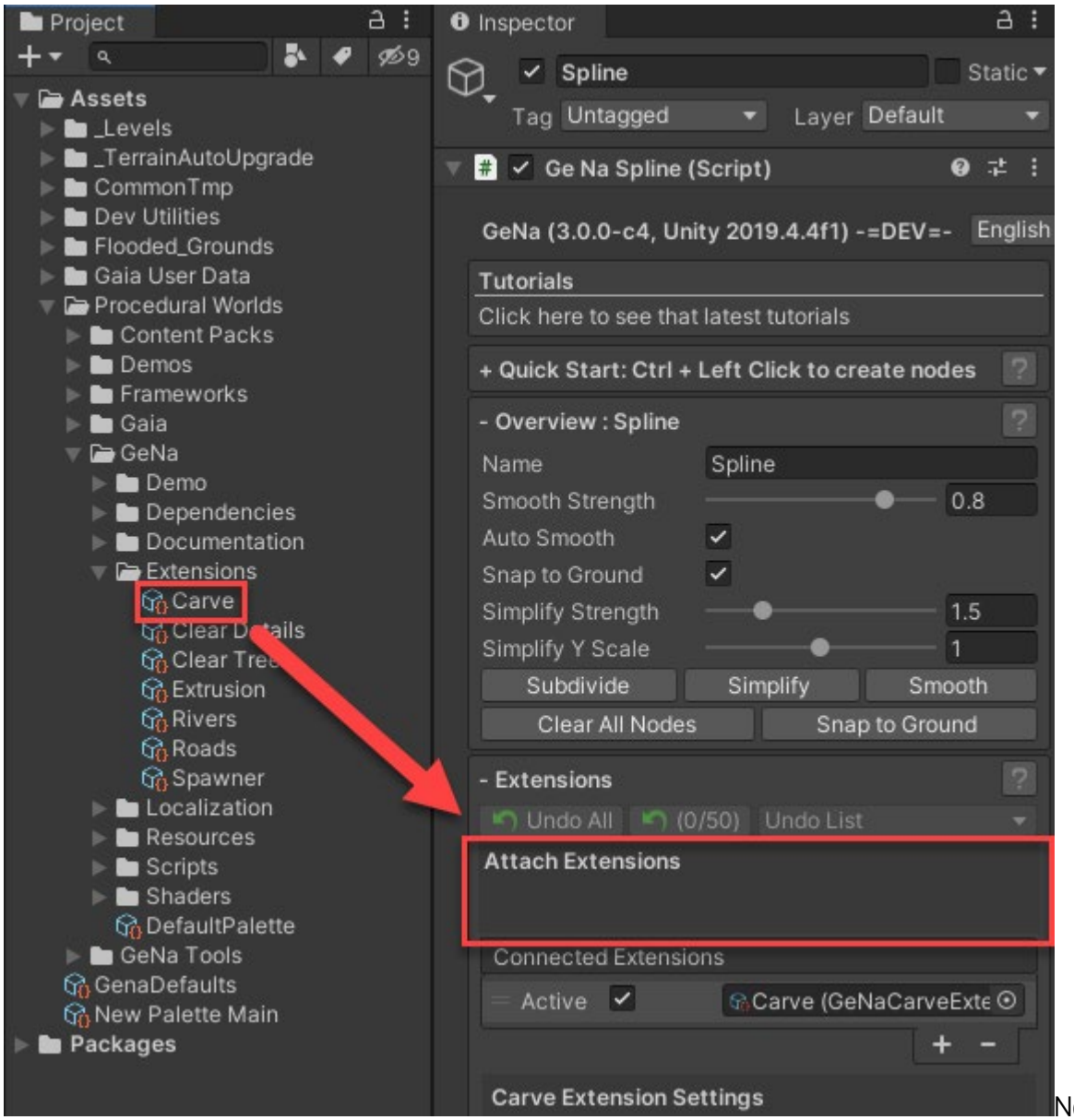


Adding Extensions

Step 1). **Using Spline Extensions**

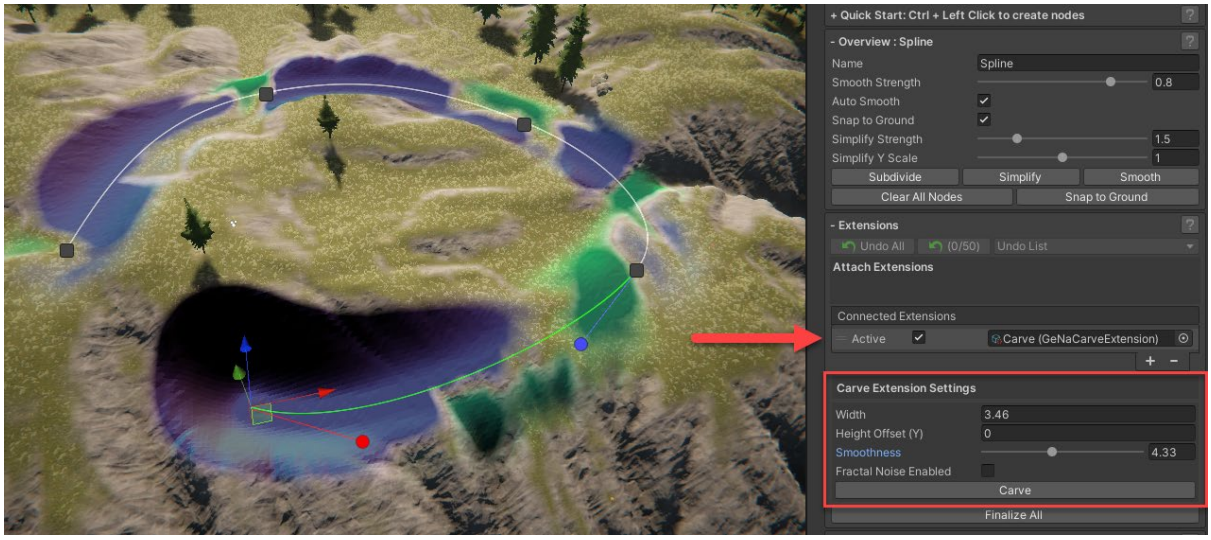
You can perform various operations along a spline depending on which 'Spline Extension' you use. Let us start by connecting a **GeNa Spawner**. These Extensions are in the form of a Scriptable Object and can be attached to a Spline.

For this example, locate the **'Carve' Extension** under **Procedural Worlds > GeNa > Extensions** then Simply Drag and Drop the Carve Extension into **'Attach Extensions'**.

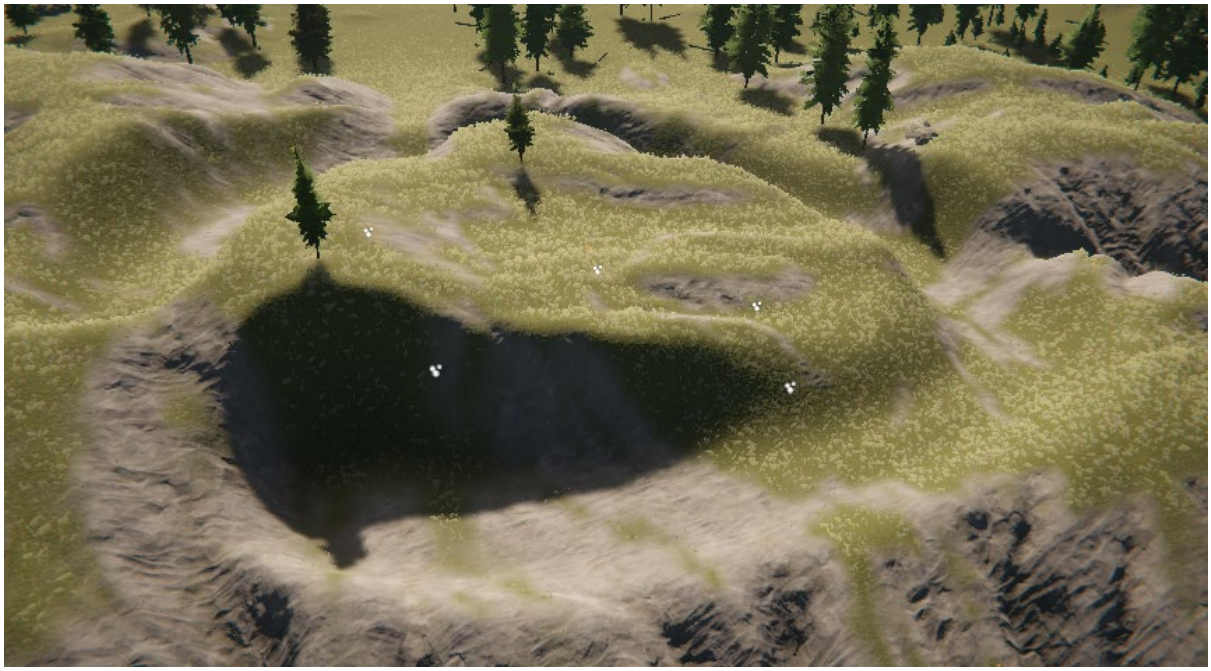


Now

Carve should be added as a **'Connected Extension'**. Here you can modify settings specific to Carve by selecting it.



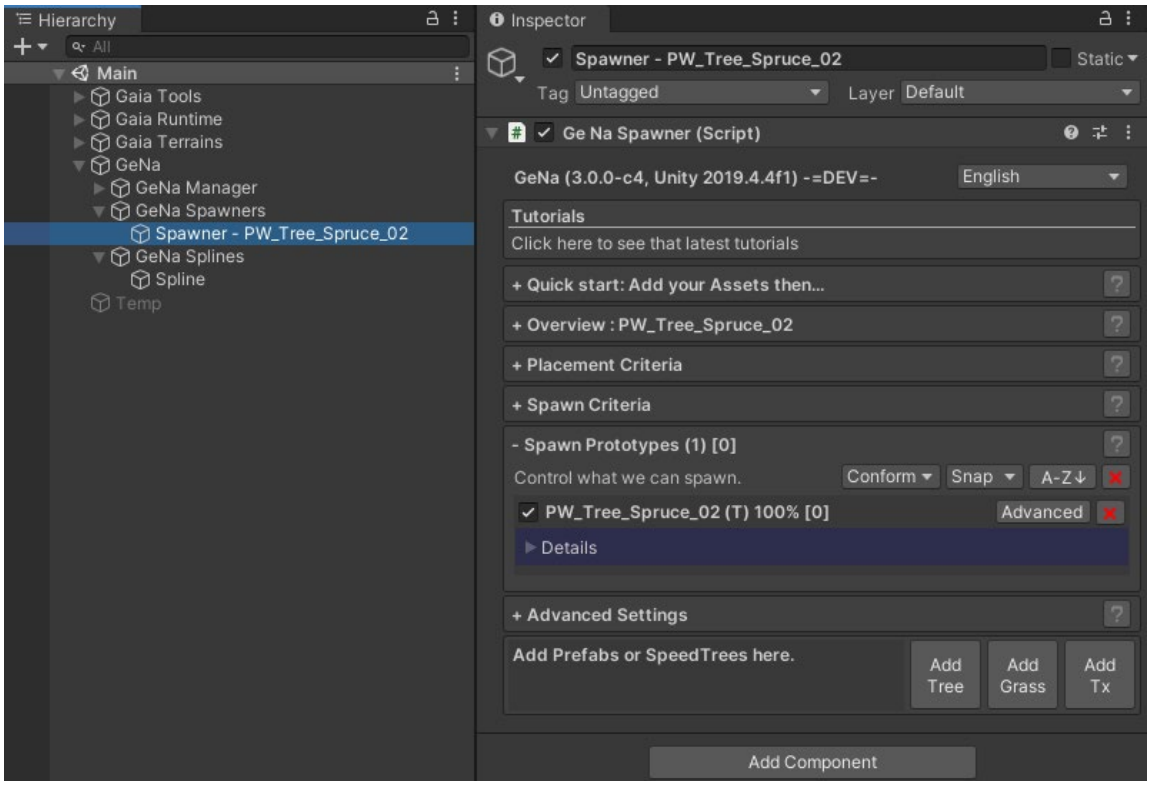
And now we can click '**Carve**' to create various embankments into our Terrain.



Spawning along Spline

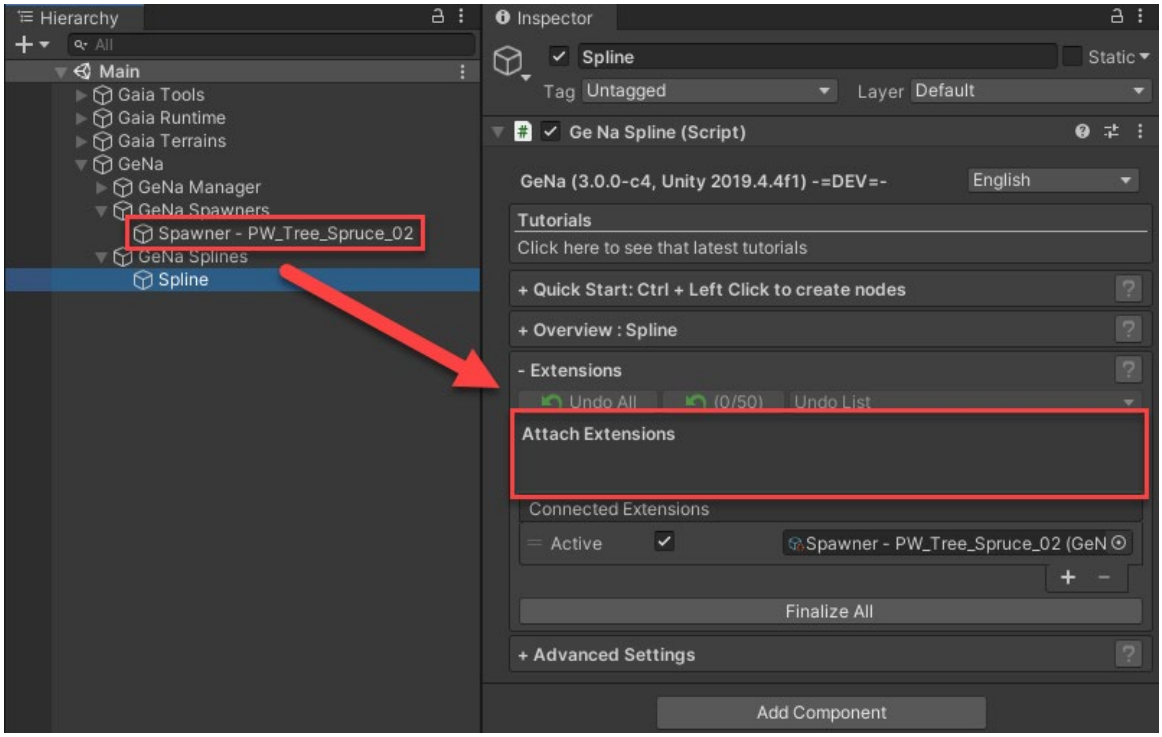
Step 1). **Create a Tree Spawner**

For this example, we will Create a GeNa Spawner that just spawns **Terrain Trees**.

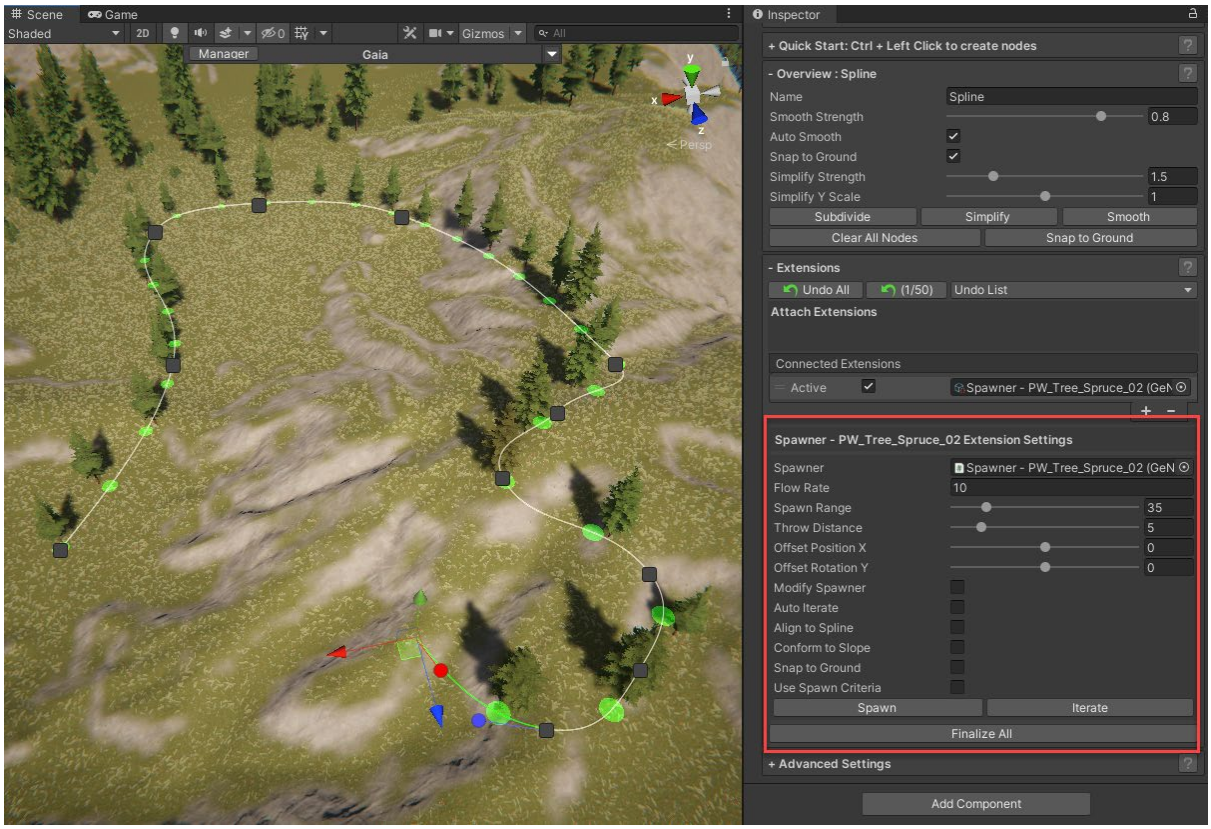


Step 2). Create a GeNa Spline

Step 3). **Click and Drag** your GeNa Spawner into the Spline's '**Add Extension**' area.

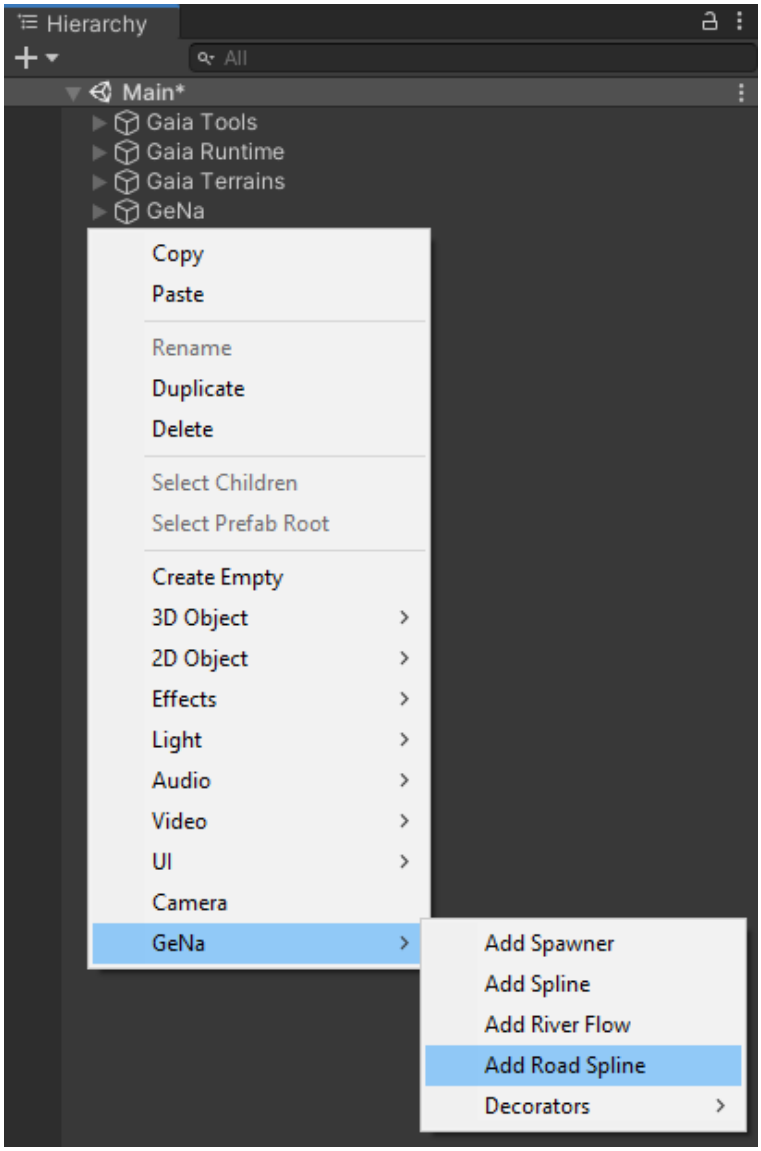


Step 4). **Select a Connected Extension to see available settings & procedures.**

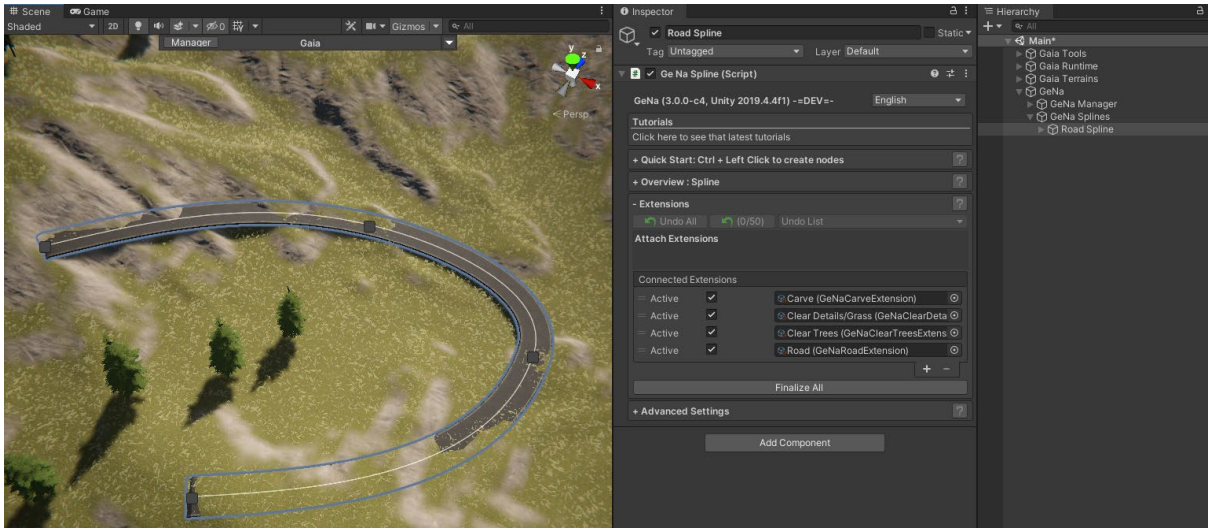


Roads

Step 1). **Right-Click** on your **Hierarchy** and select **GeNa > Add Road Spline**.



This operation will create a Road Spline with all the essential Extensions.

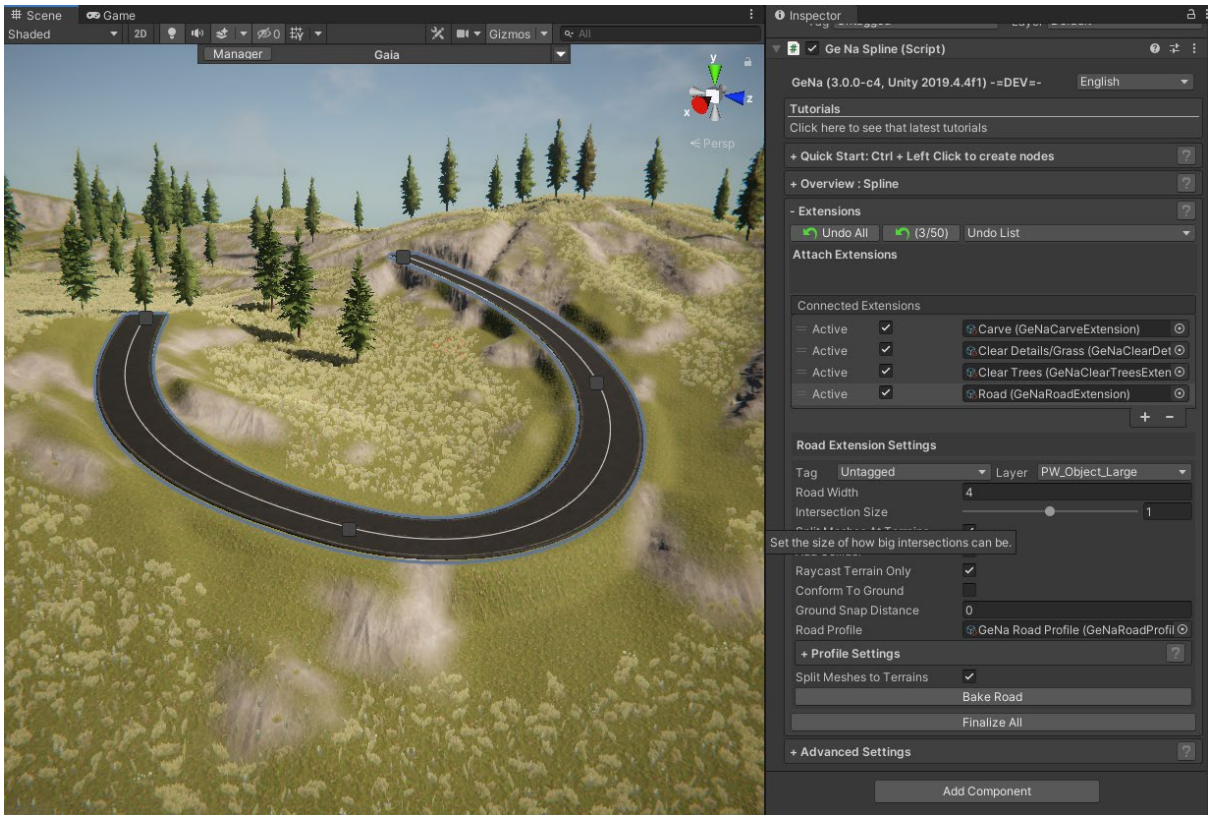


Step 2). Perform a **Carve** operation.

Step 3). Perform a Clear Trees operation.

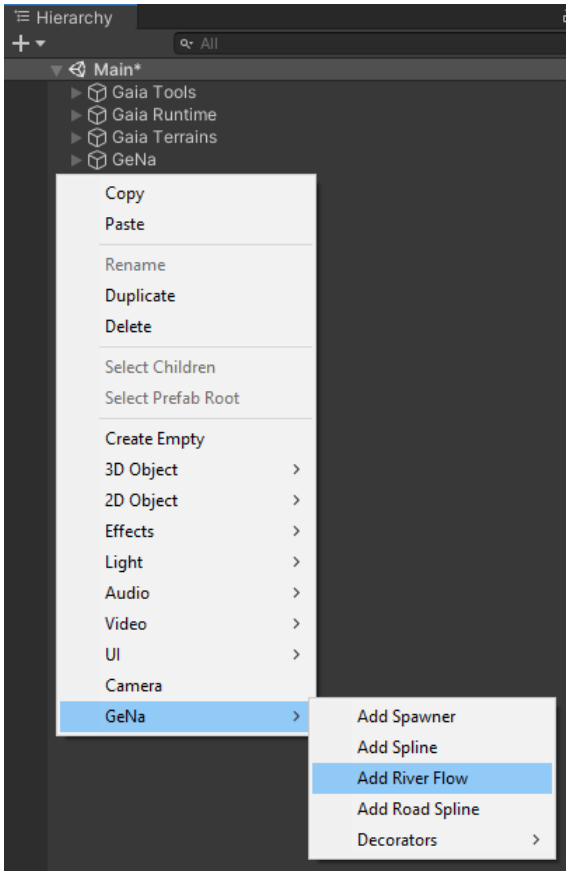
Step 4). Perform a Clear Details operation.

Step 5). Modify the Road to your liking.



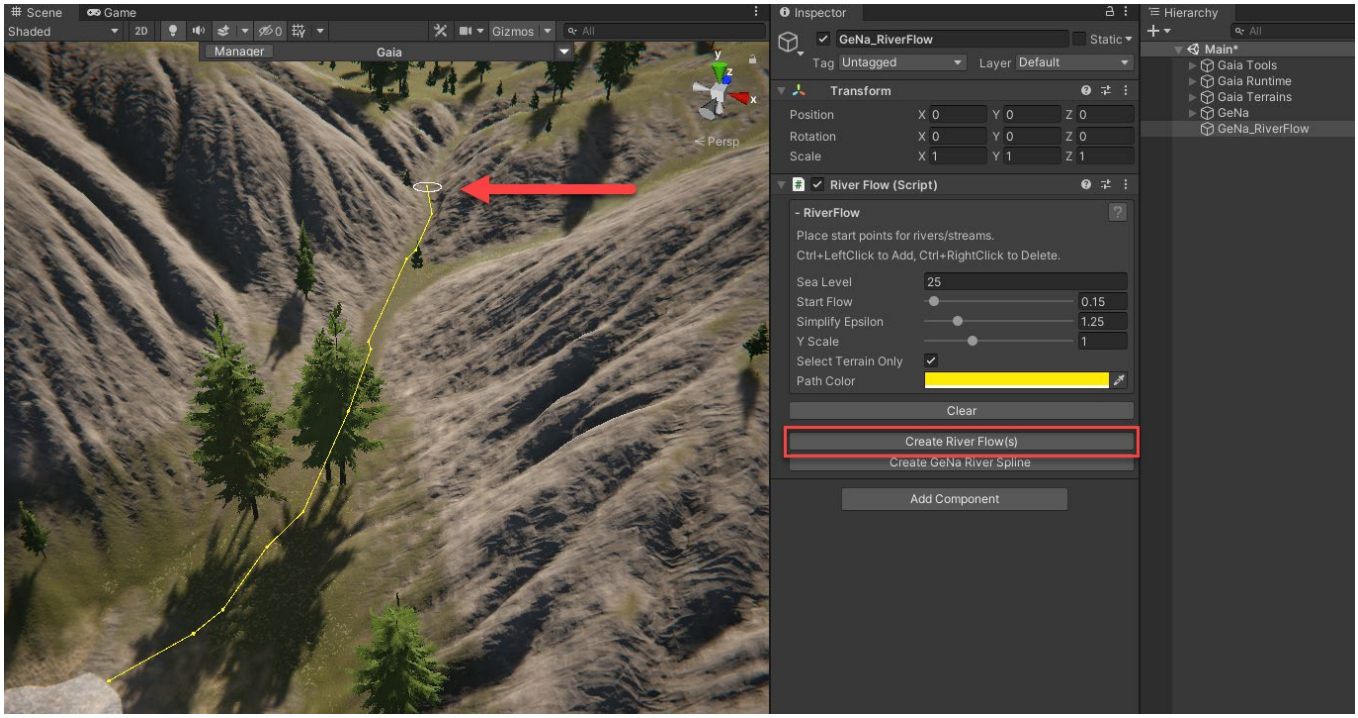
River Flow

Step 1). **Right-Click** on your **Hierarchy** and select **GeNa > Add River Flow**.



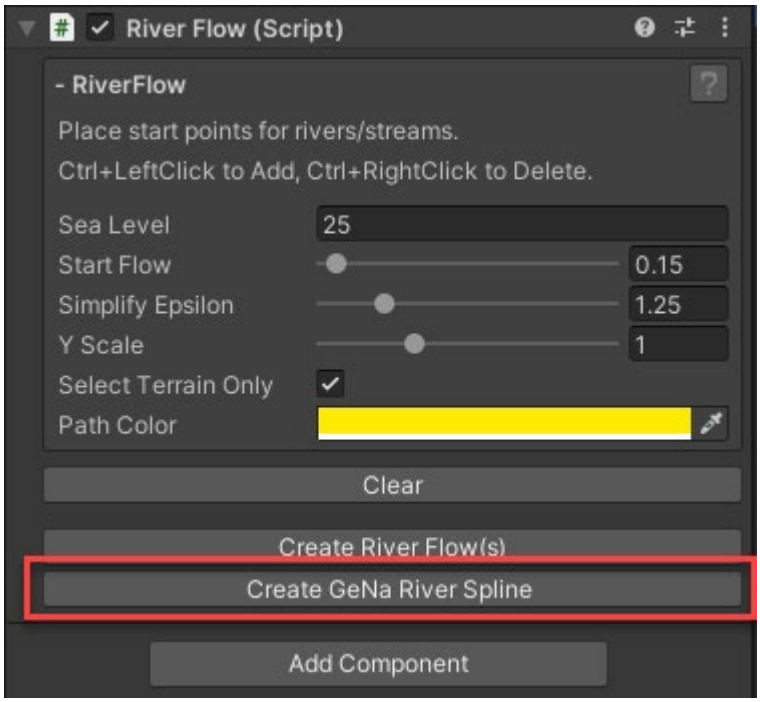
Step 2). **Sampling a River Flow**

Select the '**GeNa_Riverflow**' object in the Hierarchy and **CTRL+Left-Click** a high point on your terrain to simulate a river flow. Then click '**Create River Flow(s)**'.

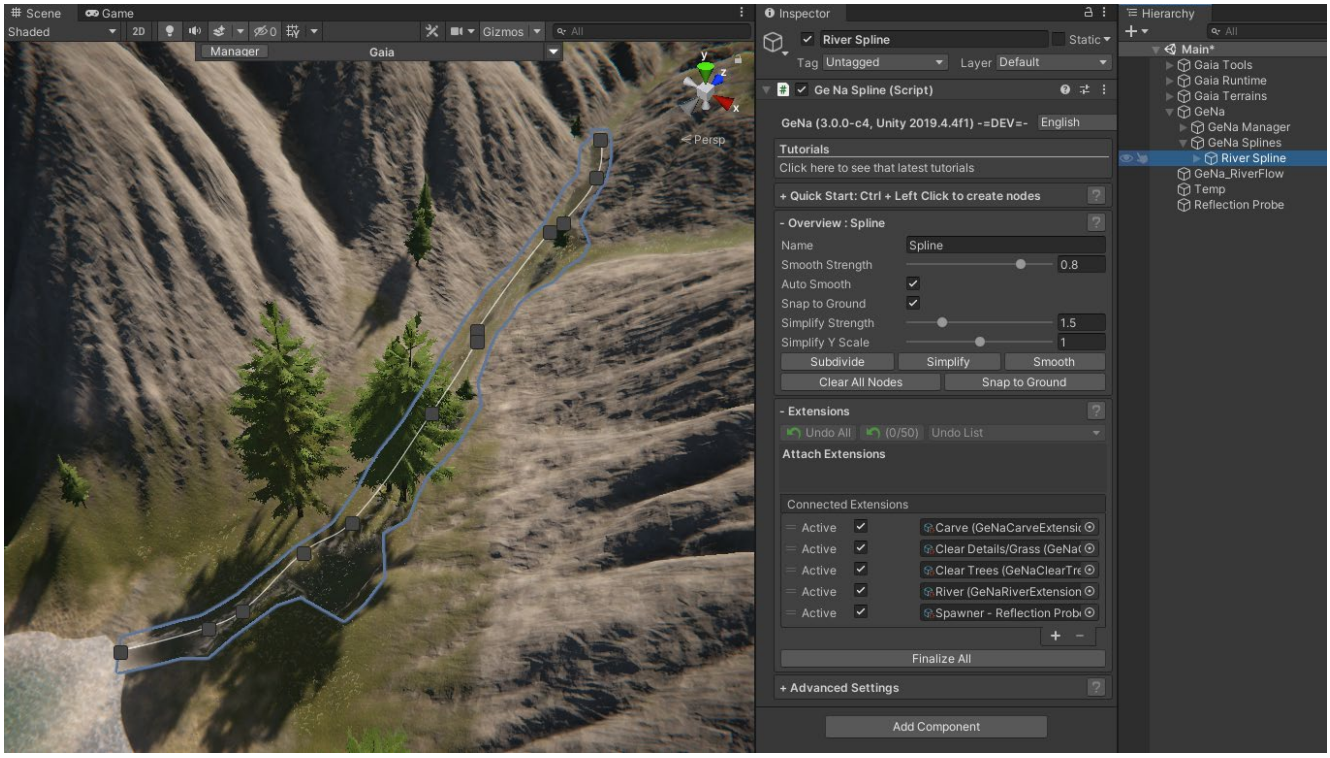


Step 3). **Generate the River Spline**

On the River Flow Script, click '**Create GeNa River Spline**'



You should now see a **River Spline** in your **Hierarchy.**



Step 4). Perform a Carve operation.

Step 5). Perform a Clear Details operation.

Step 6). Perform a Clear Trees operation.

Step 7). Perform a '**Reflection Probe**' Spawn operation.

Spawning Buildings / Villages

Step 1). Create a GeNa Spawner and add some Building / Village Prefabs.

Step 2). Configure Placement Criteria & Spawn Criteria to your liking.

Step 3). (Optional - Recommended) Create a **Prefab** out of the Spawner.

Step 4). Create a GeNa Spline and attach the Spawner to it.

Step 5). Perform a Spawn operation.

Spawning Fences / Road Lights

Step 1). Create a GeNa Spawner and add some Fences / Road Lights to the spawner.

Step 2). Set the **Throw Distance** to **zero (0)**. This will ensure the Fence is spawned at the centre of the Spawn location.

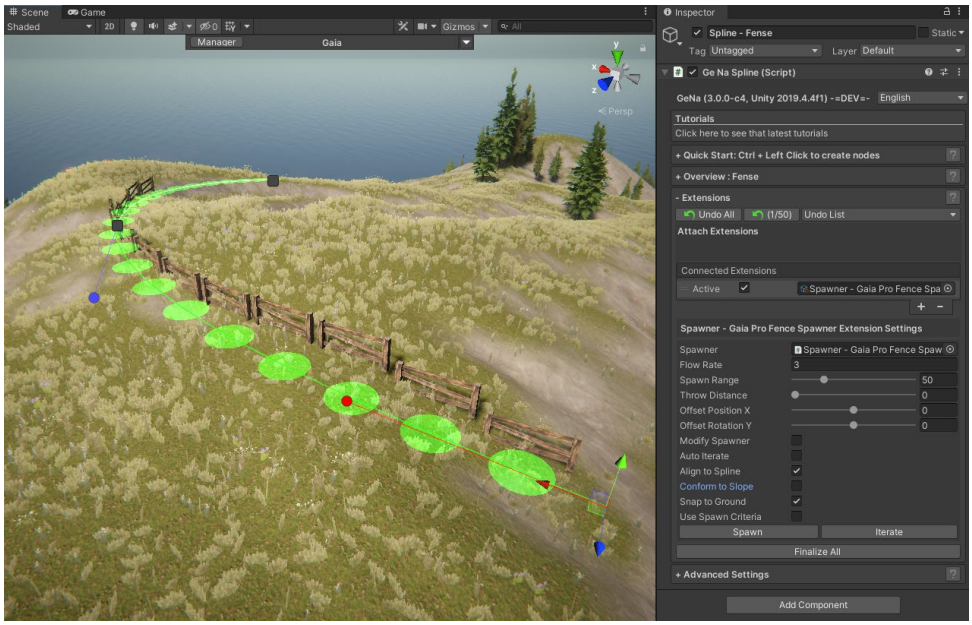
Step 3). Set the **Rotation Type** to **Mixed** and set the **Fixed Rotation** to **zero (0)**. This will allows us to control the rotation offsets on the spline later.

Step 4). Configure the rest of the settings in Placement Criteria & Spawn Criteria to your liking.

Step 5). (Optional – Recommended) Create a **Prefab** out of the Spawner.

Step 6). Create a GeNa Spline and attach the Spawner to it.

Step 7). Adjust the settings where necessary and perform a Spawn operation.



Clearing Trees

Step 1). Create a GeNa Spline.

Step 2). Attach a Clear Trees Extension to the Spline.

Step 3). Configure settings and perform **Clear** operation.

Clearing Details

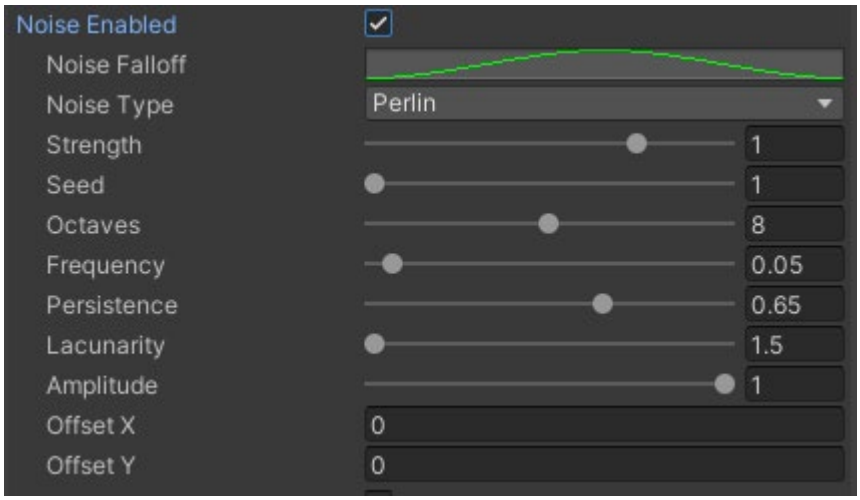
Step 1). Create a GeNa Spline.

Step 2). Attach a Clear Details Extension to the Spline.

Step 3). Configure settings and perform **Clear** operation.

Spline Noise

Certain spline extensions come with a noise system that contain a set of common properties used to apply noise along a spline.



Noise Enabled: Enables Noise Operations along the shoulder of the spline.

Noise Falloff: The curve falloff for the noise effect along the shoulder of the spline.

Mask Type: Select between Perlin, Billow, Ridged, IQ, Swiss and Jordan noise types.

Strength: Strength impact of the noise.

Seed: The seed for the noise function – the same seed will always generate the same noise for a given set of parameters.

Octaves: The amount of detail in the noise – more octaves mean more detail and longer calculation time.

Frequency: The frequency of the first octave. Smaller values create larger noise patterns.

Persistence: The roughness of the noise. Controls how quickly amplitudes diminish for successive octaves. 0…1

Lacunarity: The frequency multiplier between successive octaves. Experiment between 1.5 – 3.5.

Amplitude: The maximum extent of oscillation in height.

Offset X: The X axis offset of the noise operation.

Offset Y: The Y axis offset of the noise operation.

Ridge Offset: Ridge offset for ridged noise and swiss noise.

Warp: Warp for swiss and jordan noise.

Warp0: Warp for jordan noise.

Damp: Damp for jordan noise.

Damp0: Damp 0 for jordan noise.

Damp Scale: Damp scale for jordan noise.

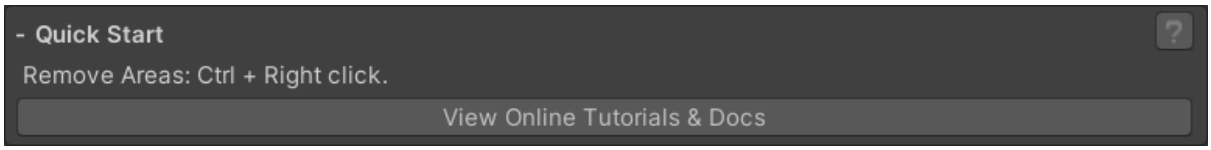
Displacement: Noise displacement.

GeNa Map Builder

The Map Builder system allows you to create various Towns, Cities and Villages in your environment.

Interface

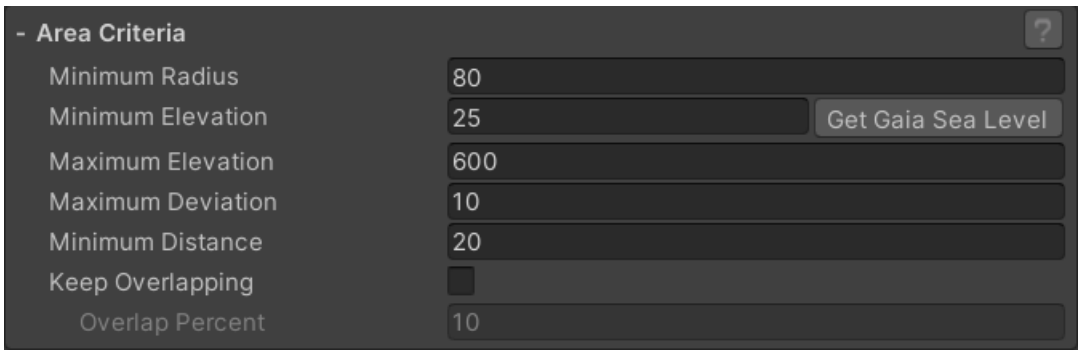
Quick Start Panel



The Quick Start panel shows the hotkeys used to control the Map Builder component.

Area Criteria Panel

Criteria used to find and filter areas.



Minimum Radius: Minimum radius of an area to be considered valid.

Minimum Elevation: The lowest altitude that will be included in an area.

Get Gaia Sea Level: Gets the Sea level from Gaia Terrain (Note: This button only appears when Gaia is present).

Maximum Deviation: The maximum difference in height that will be considered to include in an area.

Minimum Distance: The minimum distance between towns.

Keep Overlapping: Should areas that overlap be included in the result?

Overlap Percent: What percent of overlap is allowed for an area to be included?

Visualization Panel

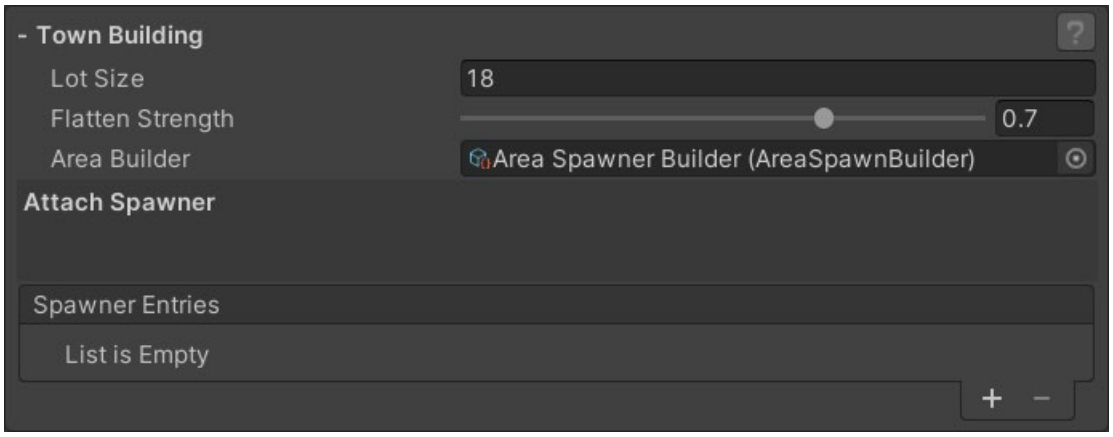
Settings for handling visualization in the Editor.

Show Elevation: Show the elevation at the cursor position.

Text Color: The color that the text should be displayed.

Town Building

Criteria for setting up the construction of Towns within the found Areas.



Lot Size: The search for where to build in areas is done in a grid pattern, and this value is the divisor or smallest grid block that a building can occupy. You can also think of it as the total width of a roadway or as the minimum lot size.

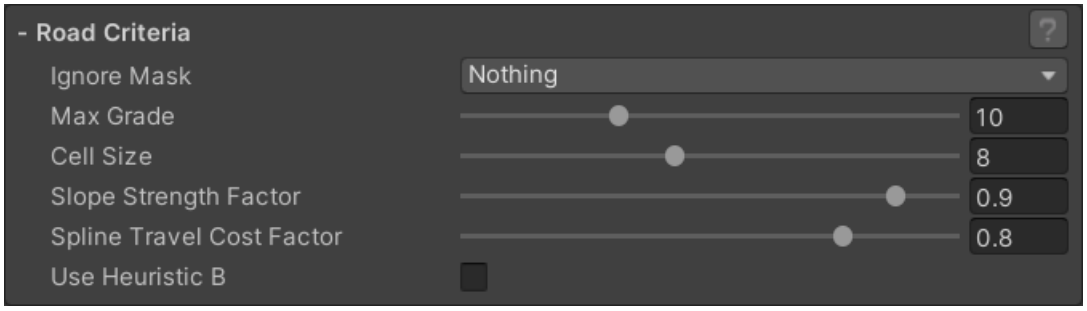
Flatten Strength: The strength of the flattening of the area for a town or city.

Area Builder: The AreaBuilder derived class that will build a town or city in each area.

Spawner Entries: A list of GeNa Spawners that can be randomly selected from to fit on to lots within a town or city.

Road Criteria Settings

Criteria used for path finding between towns.



Ignore Mask: What layers should be ignored when searching for valid moves along a path?

Max Grade: The maximum grade (height/distanceTraveled) that a road can 'breach'.

Cell Size: Step size for A* search for paths.

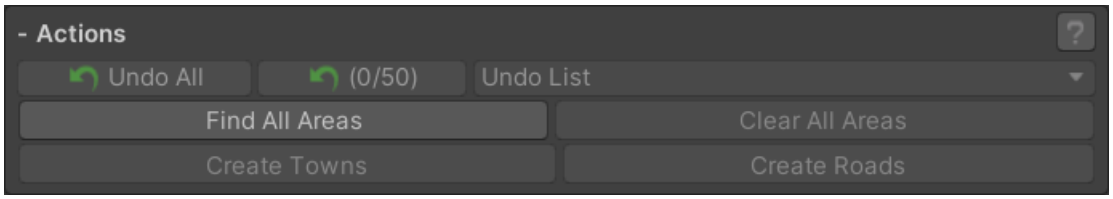
Slope Strength Factor: The amount that an increase or decrease in slope affects the cost of the path.

Spline Travel Cost Factor: Cost per meter to travel on a road. Note that it should be less than rough ground.

Use Heuristic B:

Actions Panel

All actions that can be performed with Map Builder.



Find All Areas: Searches for areas based on the criteria provided in the 'Area Criteria' panel.

Clear All Areas: Clears all the areas that have been previously found.

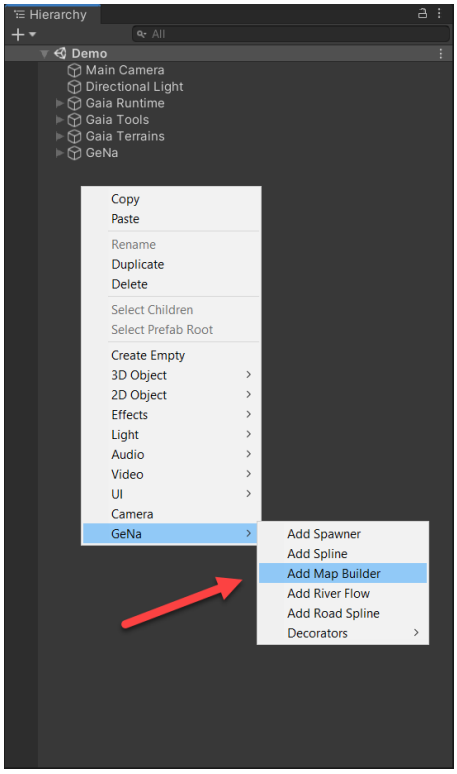
Create Towns: Creates Towns in the areas based on the 'Town Building' panel.

Create Roads: Attempts to create Roads between Towns using Path Finding from the 'Road Criteria' panel.

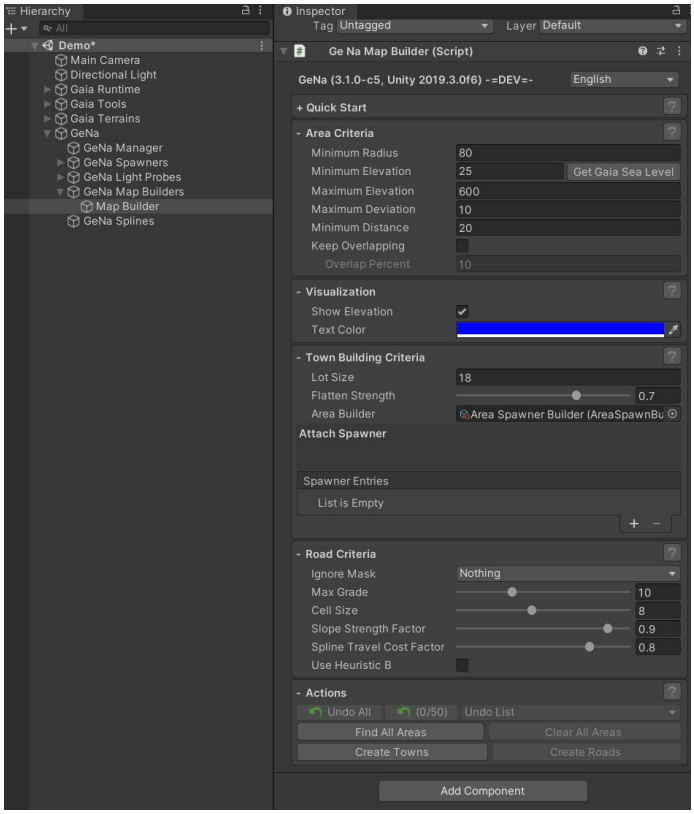
Usage

Create a Map Builder

Simply Right-Click in the Hierarchy and select **GeNa > Add Map Builder**:

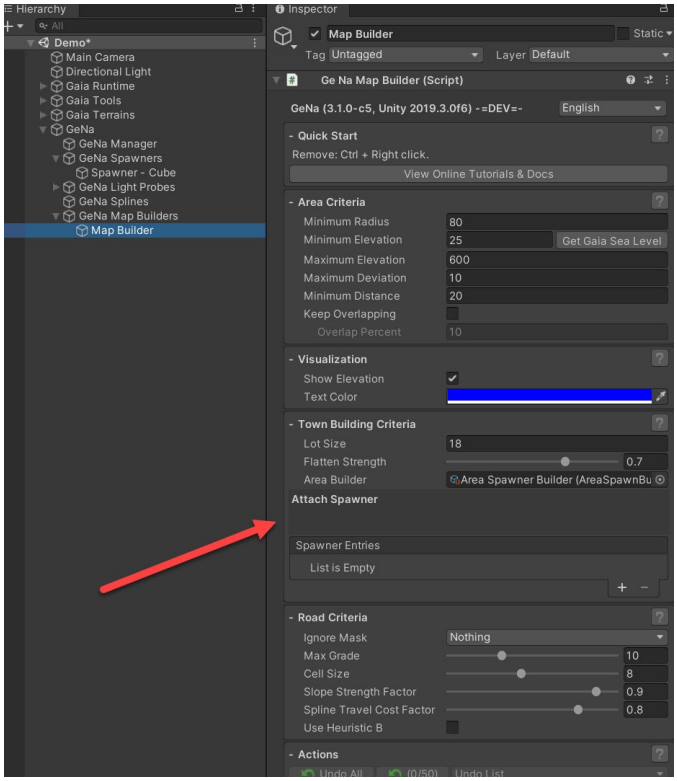


GeNa will create a new Map Builder GameObject under **GeNa > GeNa Map Builders** in the **Hierarchy** window.

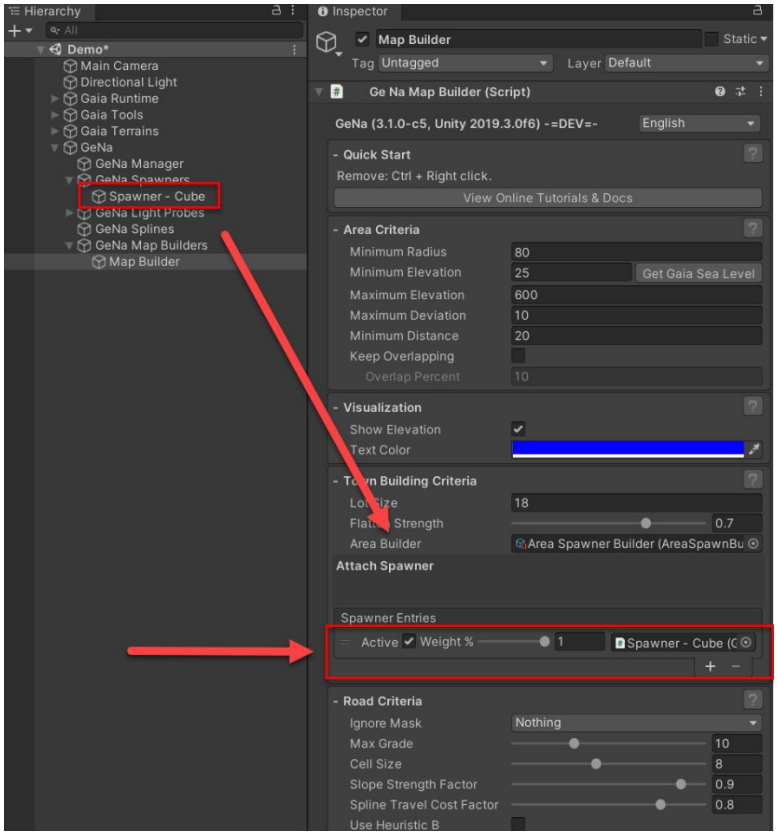


Adding GeNa Spawners

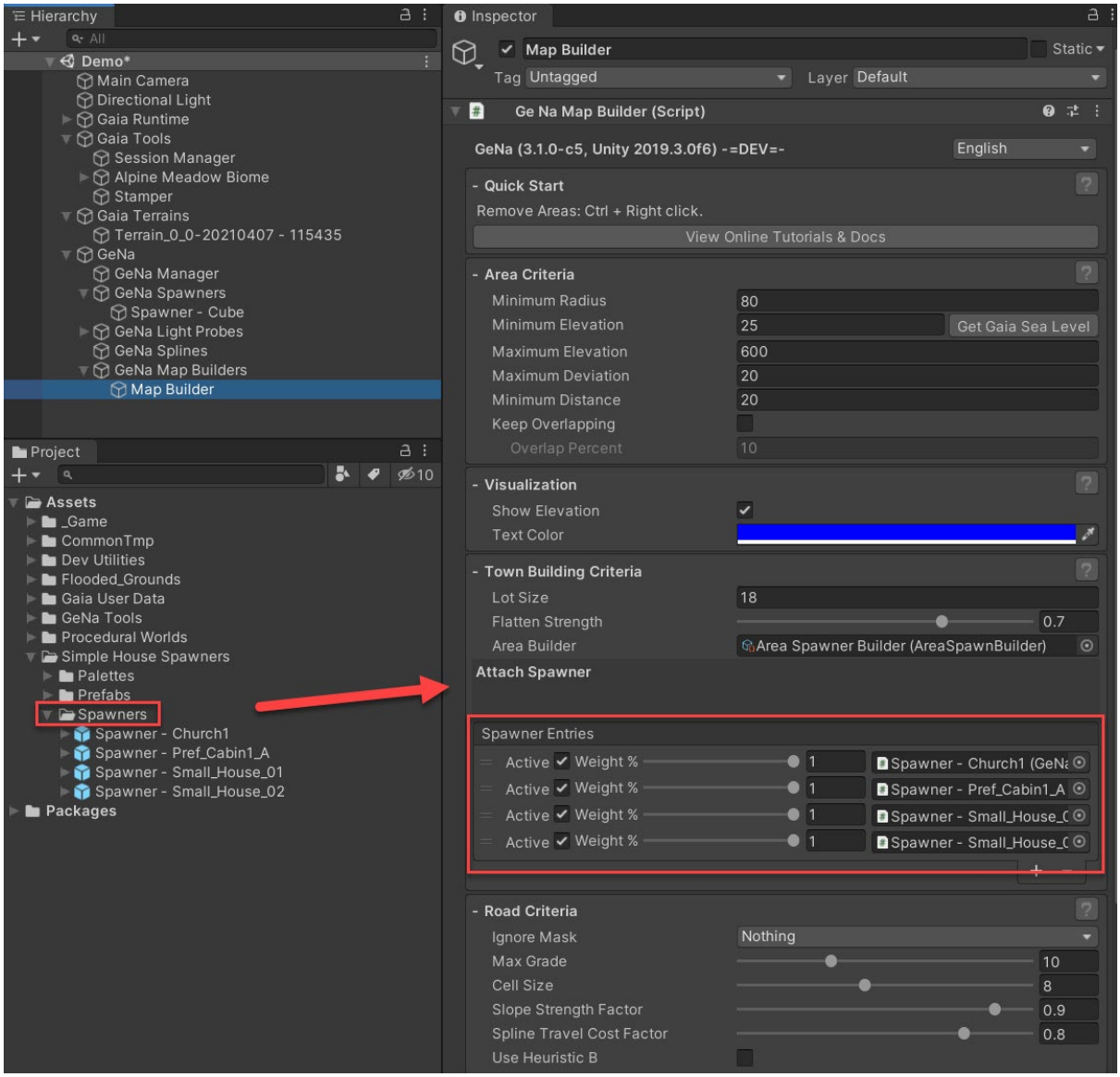
Under the **Town Building Criteria** panel, you'll find an 'Attach Spawner' drop box.



Drag and drop any GeNa Spawner(s) into this box and Map Builder will populate the Spawner Entries as follows:

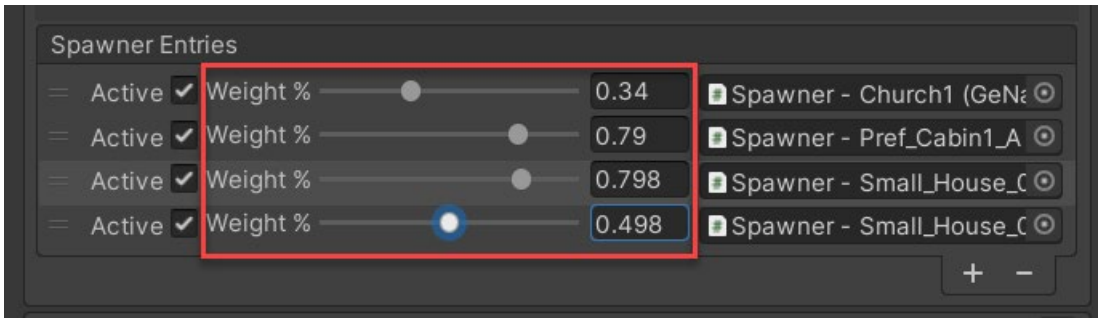


It is also possible to drag and drop a Root Folder in the Project Window to Add all of the GeNa Spawners at Once.

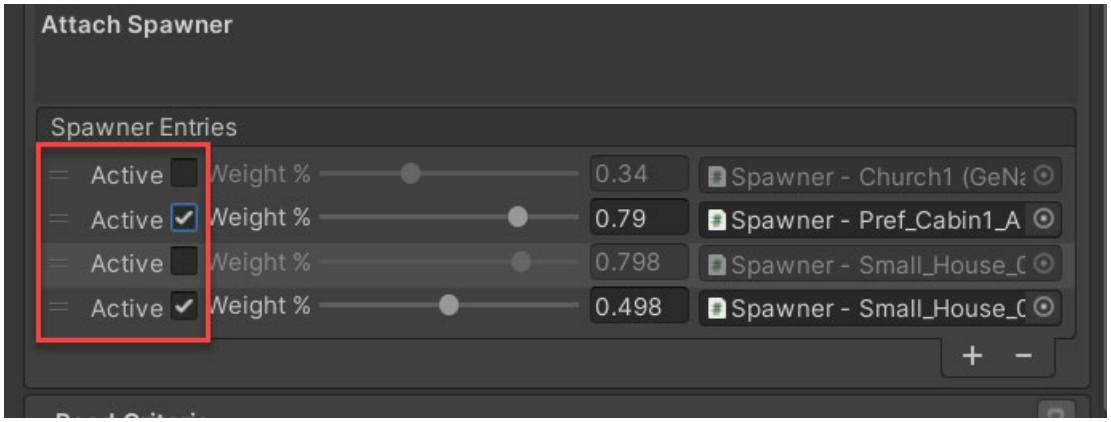

Hierarchy
All
Demo*
Main Camera
Directional Light
Gaia Runtime
Gaia Tools
Session Manager
Alpine Meadow Biome
Stamper
Gaia Terrains
Terrain_0_0-20210407 - 115435
GeNa
GeNa Manager
GeNa Spawners
Spawner - Cube
GeNa Light Probes
GeNa Splines
GeNa Map Builders
Map Builder
Project
Assets
_Game
CommonTmp
Dev Utilities
Flooded_Grounds
Gaia User Data
GeNa Tools
Procedural Worlds
Simple House Spawners
Palettes
Prefabs
Spawners
Spawner - Church1
Spawner - Pref_Cabin1_A
Spawner - Small_House_01
Spawner - Small_House_02
Packages
Inspector
Map Builder
Static
Tag Untagged
Layer Default
Ge Na Map Builder (Script)
GeNa (3.1.0-c5, Unity 2019.3.0f6) -=DEV=-
English
- Quick Start
Remove Areas: Ctrl + Right click.
View Online Tutorials & Docs
- Area Criteria
Minimum Radius 80
Minimum Elevation 25
Get Gaia Sea Level
Maximum Elevation 600
Maximum Deviation 20
Minimum Distance 20
Keep Overlapping
Overlap Percent 10
- Visualization
Show Elevation
Text Color
- Town Building Criteria
Lot Size 18
Flatten Strength 0.7
Area Builder Area Spawner Builder (AreaSpawnBuilder)
Attach Spawner
Spawner Entries
Active Weight % 1 Spawner - Church1 (GeNa
Active Weight % 1 Spawner - Pref_Cabin1_A
Active Weight % 1 Spawner - Small_House_C
Active Weight % 1 Spawner - Small_House_C
- Road Criteria
Ignore Mask Nothing
Max Grade 10
Cell Size 8
Slope Strength Factor 0.9
Spline Travel Cost Factor 0.8
Use Heuristic B

Configuring Spawner Weighting

GeNa Spawners can be weighted depending on how often you wish each Spawner to be used.

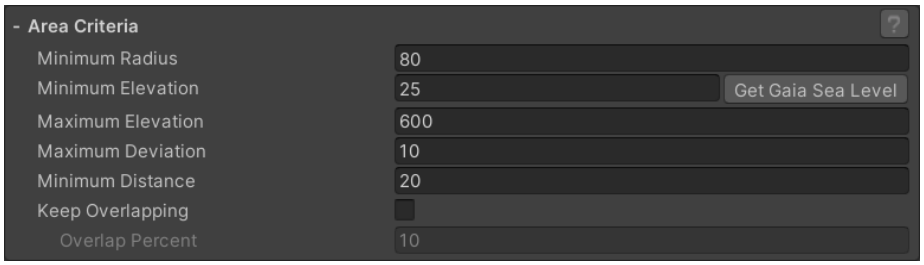


You can also Activate / Deactivate Specific Spawners before performing any actions with Map Builder.

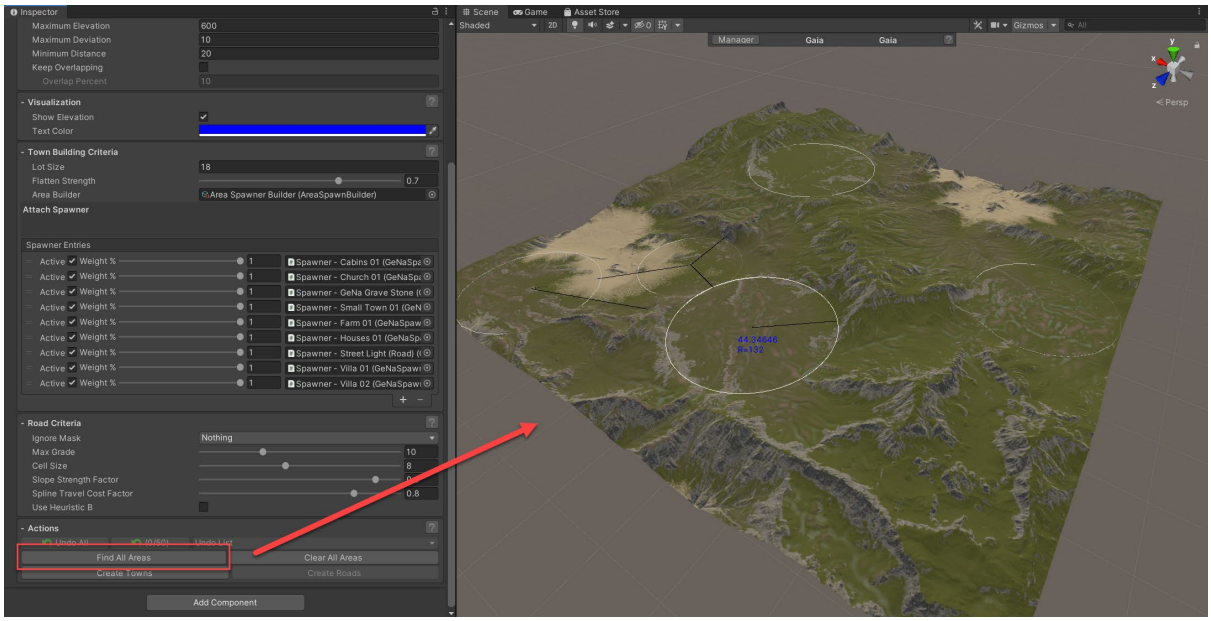


Finding All Areas

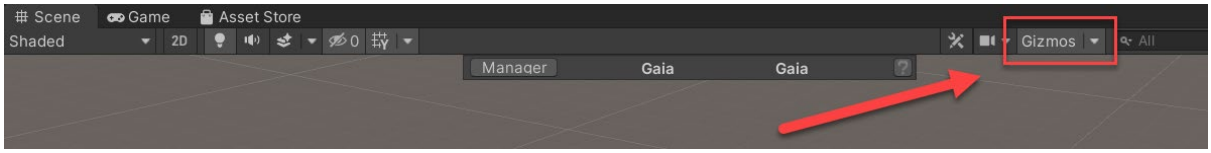
Map Builder needs to know where to generate Towns in your Environment. The criteria for this can be found in the '**Area Criteria**' panel.



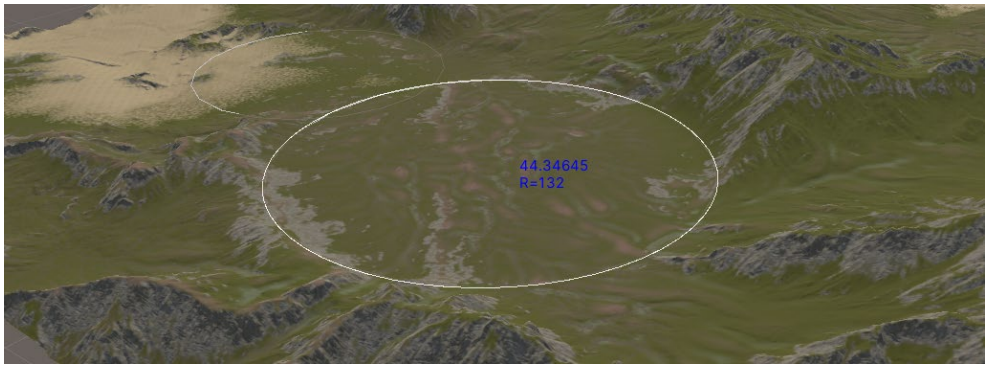
After setting up your Area Criteria, simply click '**Find All Areas**' in the '**Actions**' panel to get Map Builder to scan your environment for them.



The areas are highlighted in your Scene (provided you have **Gizmos** enabled)



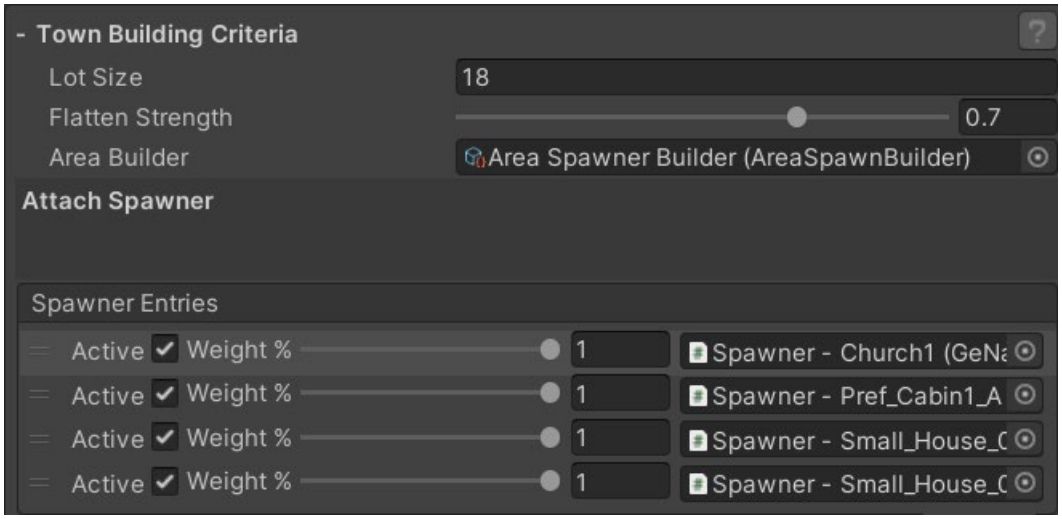
You can also easily Remove unwanted Areas by holding Ctrl + Right Click on the specified area.



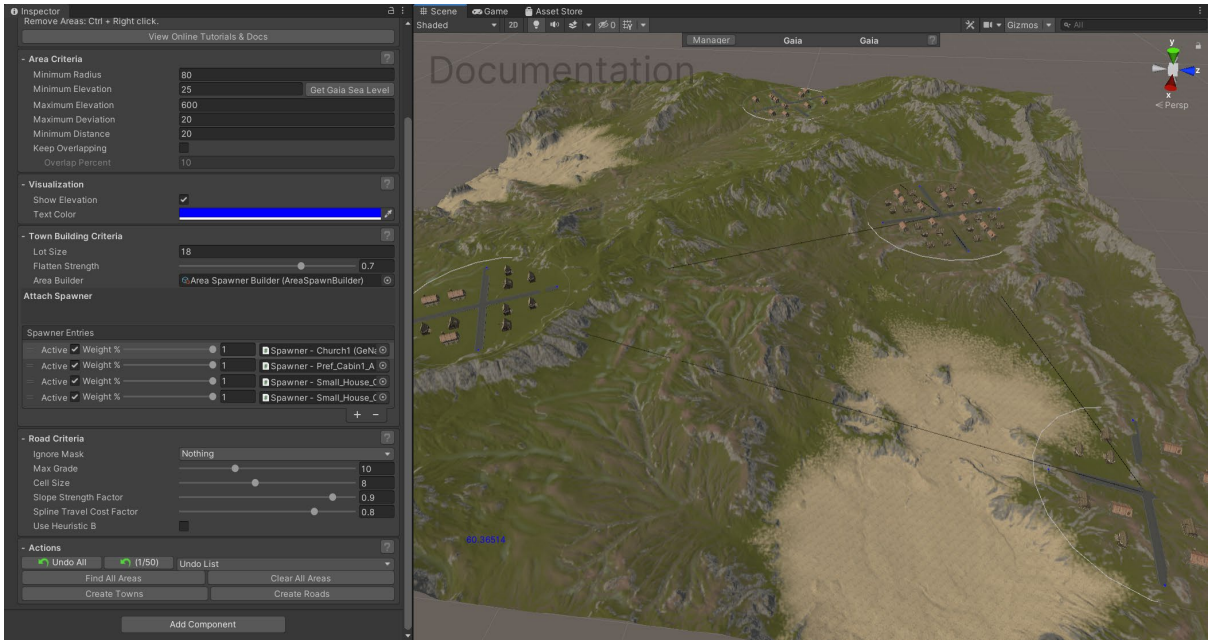
Creating Towns

Now that Map Builder has identified the areas, we can now Spawn Towns!

You can modify the output by changing settings in the '**Town Building Criteria**' panel.



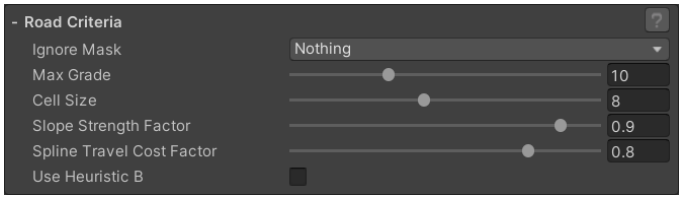
Then click '**Create Towns**' in the '**Actions**' panel to start this process.



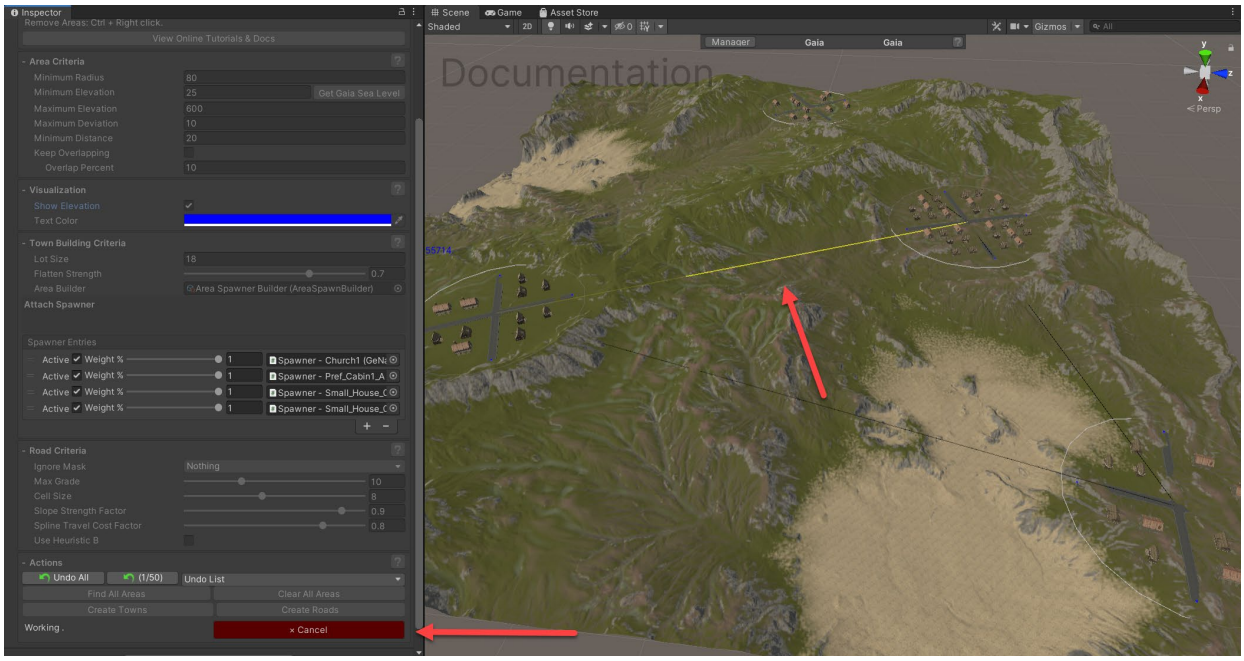
Map Builder has successfully placed our buildings in the correct lot sizes.

Creating Roads

To create Roads, you can modify the '**Road Criteria**' panel.



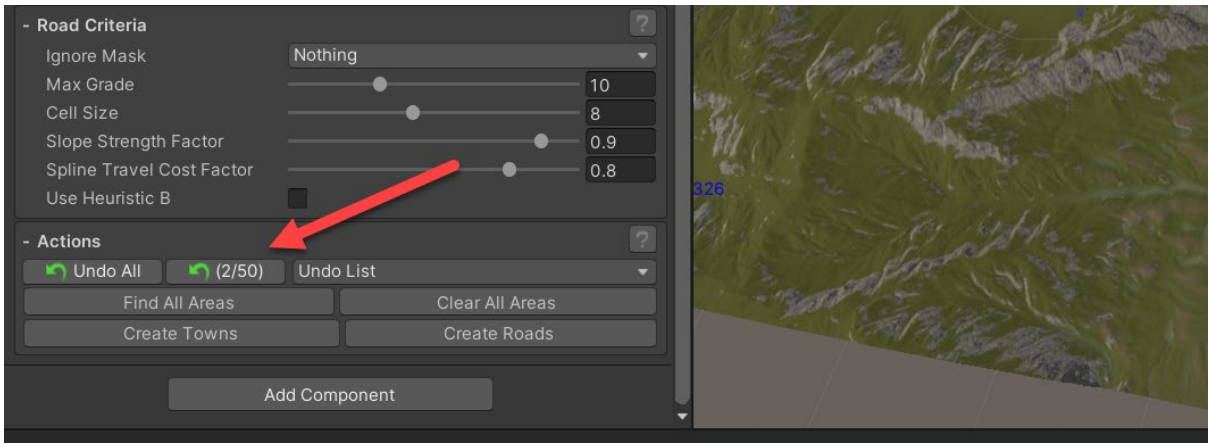
Followed by the '**Create Roads**' button in the '**Actions**' panel.



Map Builder will determine where to connect the roads of each Town by using Path Finding in your environment. The current path being generated is highlighted yellow. You can also cancel this process by clicking the red '**Cancel**' button in the inspector.

Perform Undo Operations

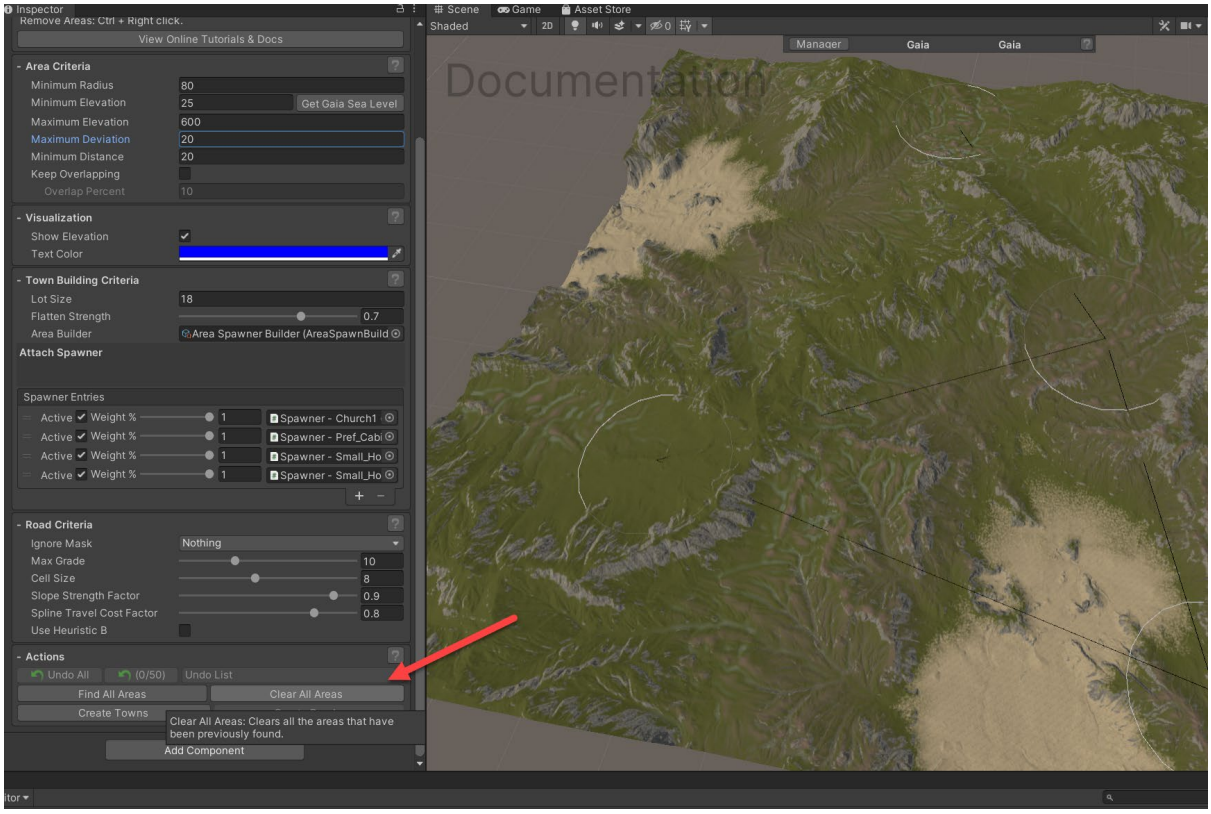
You can also undo all operations by performing the Undo operation here.



This will remove all the Flatteners, Towns, Roads, etc.

Clear All Areas

You may also wish to clear all the areas that were found by Map Builder in order to Find All Areas again.



Components

GeNa Growth Script

As a fun value add, Gena comes with a simple but efficient growth script that can be placed on things like trees, and then used either at design time or run time.

To demonstrate let's perform the following steps:

1. Physically drag a speed tree into your scene.
2. Attach the "GenaGrowthScript.cs" to your tree and set the growth settings.
3. Save the tree as a prefab by dragging it into your assets folder.
4. Delete the tree from your scene.
5. Set the x, y, and z values on the prefab to zero.
6. Create a spawner and drop the new tree prefab onto it.
7. Update the prototype and un check "Conform Slope" under the detail folder.
8. Spawn a bunch of trees into your scene.
9. Click play and watch them grow ☺
10. Optionally, set these up on a runtime spawner and spawn growable trees at runtime.

For a video example please see:
https://www.youtube.com/watch?v=mFhECi_6qak&index=5&list=PLckEMv5tz0E6Gw6PlN1vP0c2M-ZRMG63F

Mouse & Keyboard Controls

GeNa is controlled by both the keyboard and the mouse, and it's the combination of these and the dynamic way that GeNa interprets where you click that makes it so powerful.
You can modify these defaults by changing the settings in the GeNa Defaults file.

Shift + Left Click

Visualise the environment at that location.

Ctrl + Left Click

Run a spawn iteration at the location.
Ctrl + Left Click + Drag

Paint spawn iterations at the location when Paint mode selected.

Shift + Ctrl + Left Click

Run a global spawn iteration based on the scene interpretation of that location i.e. find similar locations across the entire scene and run a spawner iteration on them.

Ctrl + Backspace

Delete all instances of the resources managed by this spawner from the scene. USE WITH CAUTION as it will delete every instance of the same resource regardless of whether it was placed by this spawner or not.

Shift + Left Click & Drag

Modify the rotation of the prototype. Only active when Draggable Rotation is enabled in the Advanced Settings panel.

Shift + Scroll Wheel

Change the range of the spawner.

Ctrl + Scroll Wheel

Change the instances of the spawner.
Ctrl + Left, Right, Up, Down Arrow

Move the position of the last prefab spawned.

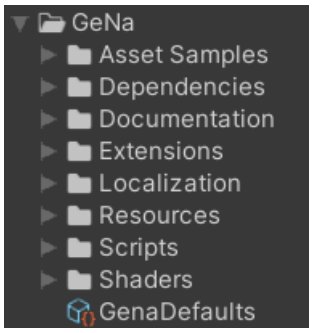
Shift + Ctrl + Up, Down Key

Change the Y axis height of the last prefab spawned.

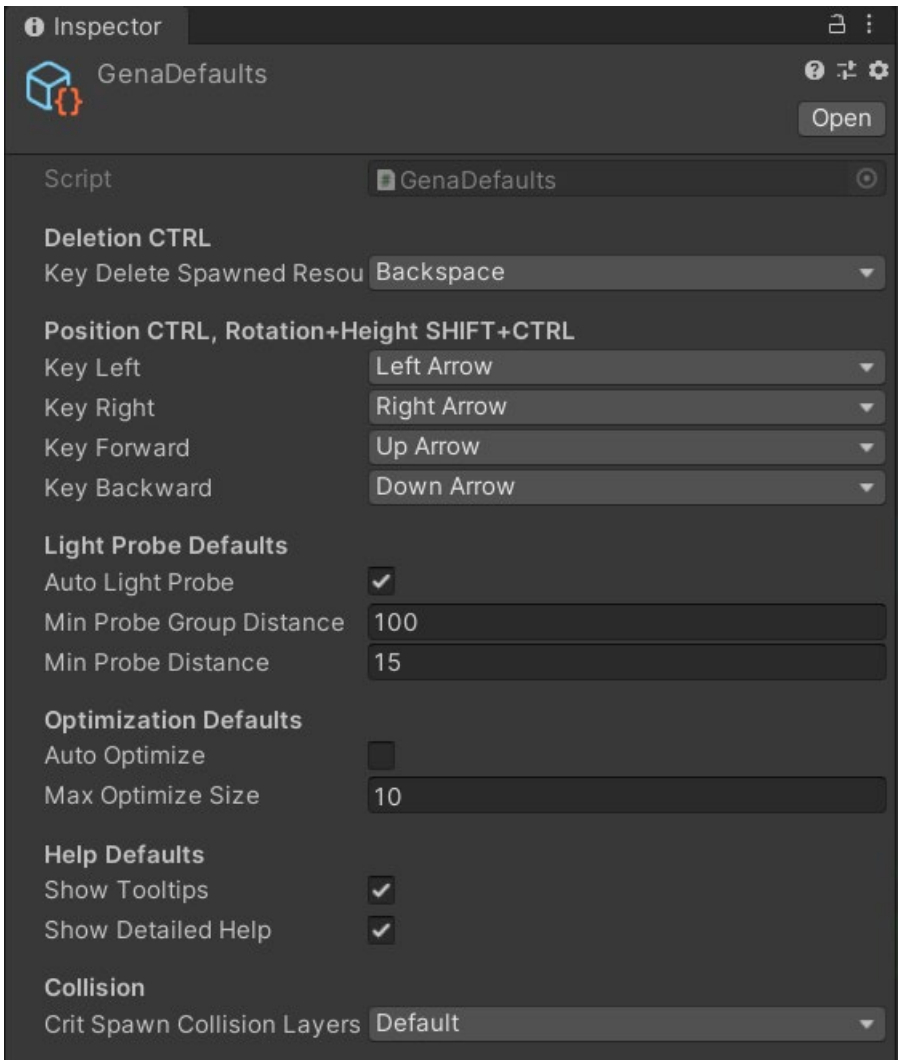
Shift + Ctrl + Left, Right, Up or Down Keys

Change the Y axis rotation of the last prefab spawned.

Defaults File



Located in the GeNa directory, the defaults file allows you to change the defaults that are added to every new spawner. There can be only one of these files per project, and it you mess this file up then just delete it and GeNa will recreate it with factory defaults.



The settings here are reflected in the way the spawner behaves.