Table of Contents

Varia is a Unity addon for easily adding random variations and recursive systems to your game. Get it here.

Varia works by simply attaching simple components to your objects. These components can be combined to achieve a variety of effects.

For help, join my Discord server.

To get started with Varia 1.1.0, see the introduction.

Introduction

Varia is a Unity addon for adding random variation

Let's do a simple example to demonstrate how Varia works. Create a new scene, and add a cube, then a VariaRandomRotation component to it.

When you hit play, the cube will be randomly rotated about the y-axis by a random amount.

That's it! The majority of Varia's basic components all work similarly - they make a small randomized change to a game object when it starts, i.e. when loaded into the scene or instantiated from a prefab.

By annotating your game objects with Varia components, you can introduce random variation to stop your game feeling samey and repetitive. Once you delive into the more advanced features of Varia, you can generate complex objects and scenes without writing a line of code.

Here's a few ideas of what you can do with Varia components:

- Randomly tint and scale NPCs to make a crowd look less homogenous
- Pick between several alternative enemies for the player to fight.
- Swap out alternative sprites to make placing scenery easy.
- Add special powerups that are only only appear occasionally.
- Browse the supplied samples for a showcase of some of Varia's features.

Once you've mastered the basic components, you can learn how to use the more advanced features:

- Use VariaPreviewer to quickly prototype and visualize more complex randomizations directly in the editor.
- Use the condition system to control if the changes are applied at all.
- Instantiate recursive prefabs to make elaborate Lindenmayer systems - great for trees and plants

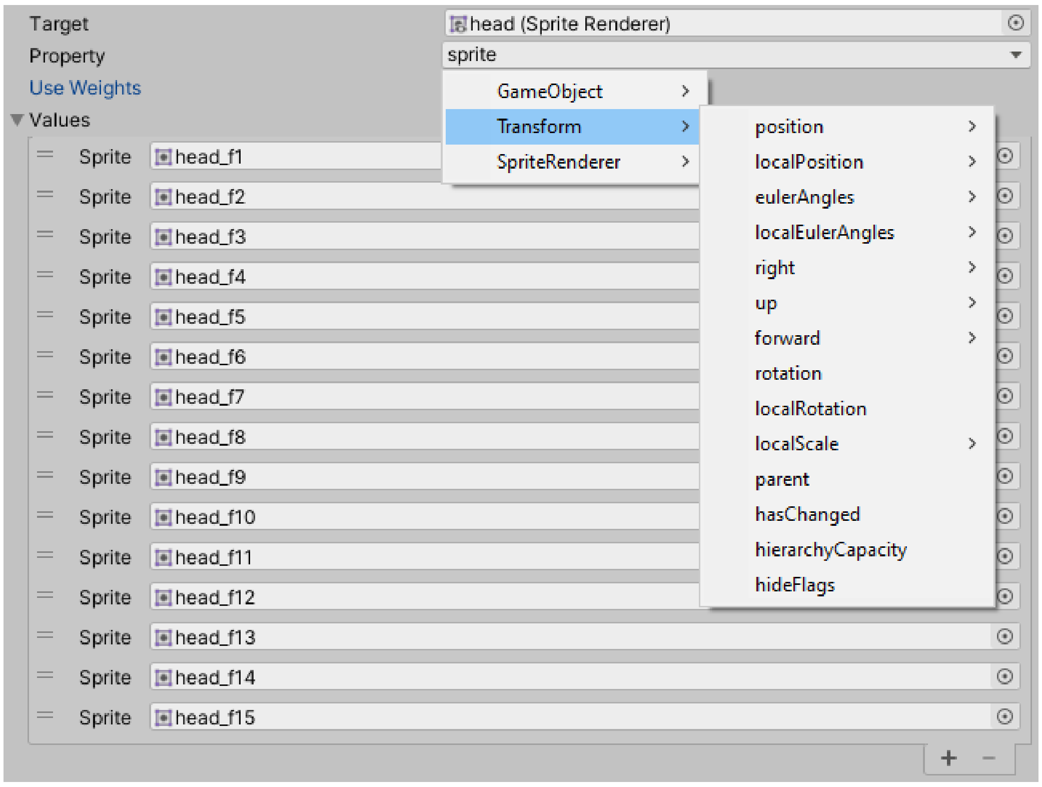
Varia Random Value

`VariaRandomValue` is an especially powerful component of Varia. It does one of the most fundamental operations of procedural generation - it picks a random value from a list of values, then assigns it to a given property.

Usage

First, use the property dropdown to select the specific component and property that should be changed.

Then fill in the values array with possible values. Optionally, Weights can be set to alter the probability of chosing each item.



Target: head (Sprite Renderer)
Property: sprite
Use Weights
Values
Sprite head_f1
Sprite head_f2
Sprite head_f3
Sprite head_f4
Sprite head_f5
Sprite head_f6
Sprite head_f7
Sprite head_f8
Sprite head_f9
Sprite head_f10
Sprite head_f11
Sprite head_f12
Sprite head_f13
Sprite head_f14
Sprite head_f15

`VariaRandomValue` works with most primitive types (like `int`, `float`), Unity built in types (like `Vector3`) and with GameObjects and components. A few other types work, but aren't supported in the Editor Inspector. It can also set material property blocks.

Warning

VariaRandomValue uses reflection to edit properties. This can cause issues in Ahead-of-time compilation build outputs. The best way to fix this is to add a C# script somewhere in your project that references the properties you want to target, forcing Unity to include them in the final output.

Settings

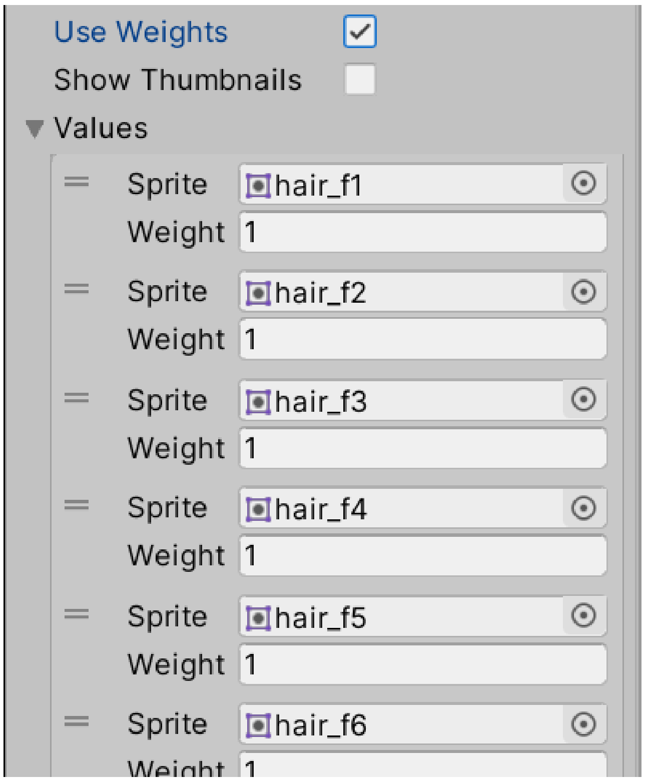
Target / Property

The target is the Unity component that contains the property you want to edit. It defaults to the relevant component of the game object that the VariaRandomValue component is on, called Self.

Property is a string that records which property on the component to change. It follows a special syntax, but it'll be set for you automatically by picking the property from the dropdown.

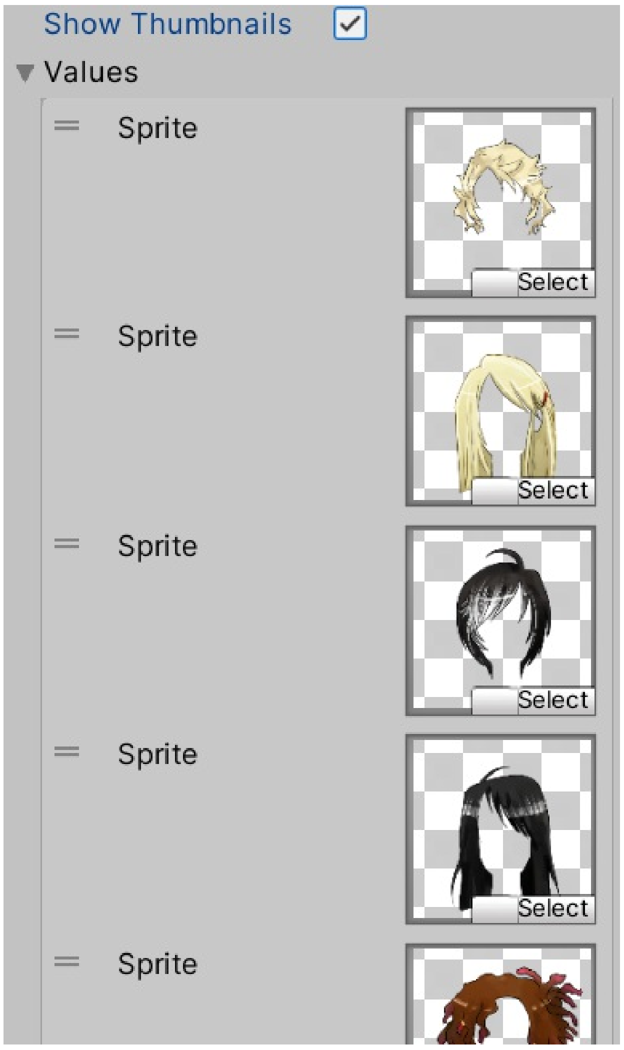
Use Weights

By default, each value in the list is equally likely to be picked. If you turn on use weights, then you can set extra values to make some values more or less likely. Each value is picked with frequency proportional to the supplied weight.


Use Weights ✓
Show Thumbnails
Values
Sprite hair_f1
Weight 1
Sprite hair_f2
Weight 1
Sprite hair_f3
Weight 1
Sprite hair_f4
Weight 1
Sprite hair_f5
Weight 1
Sprite hair_f6
Weight 1

Show Thumbnails

Toggles an alternative view of the list showing thumbnails for each value (if available).


Show Thumbnails ✓
Values
Sprite
Sprite
Sprite
Sprite
Sprite

Conditions

See Conditions.

Varia Destroy / Keep

VariaDestroy isn't random, it simply destroys the object it is attached to. It automatically comes with a condition that causes it to only run 50% of the time.

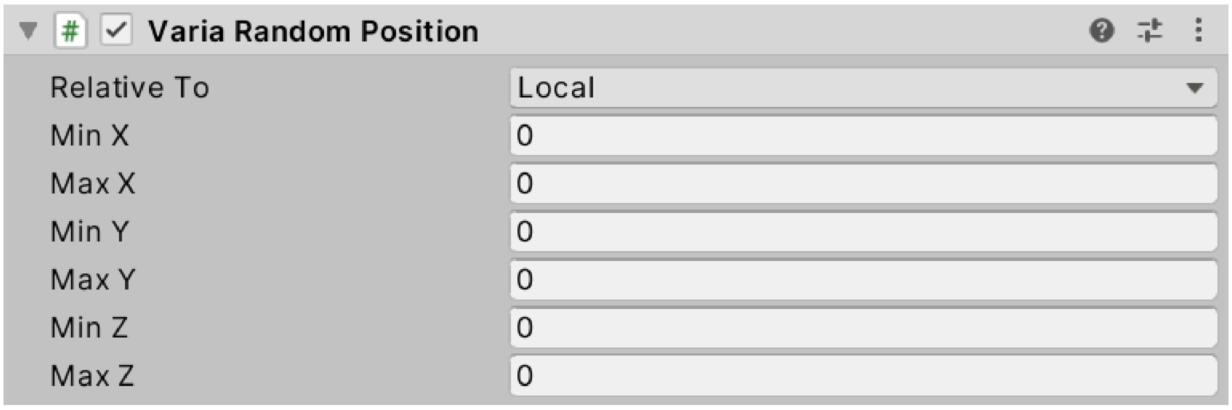
VariaKeep works the same, except it destroys the object only if the conditions fail.

Destroy and Keep can be used for randomly removing content from the game. If you want to randomly add things, use VariaInstantiate.

Varia Random Position / Rotation / Scale

This trio of components respectively randomize the position, rotation and scale of transform.

Position



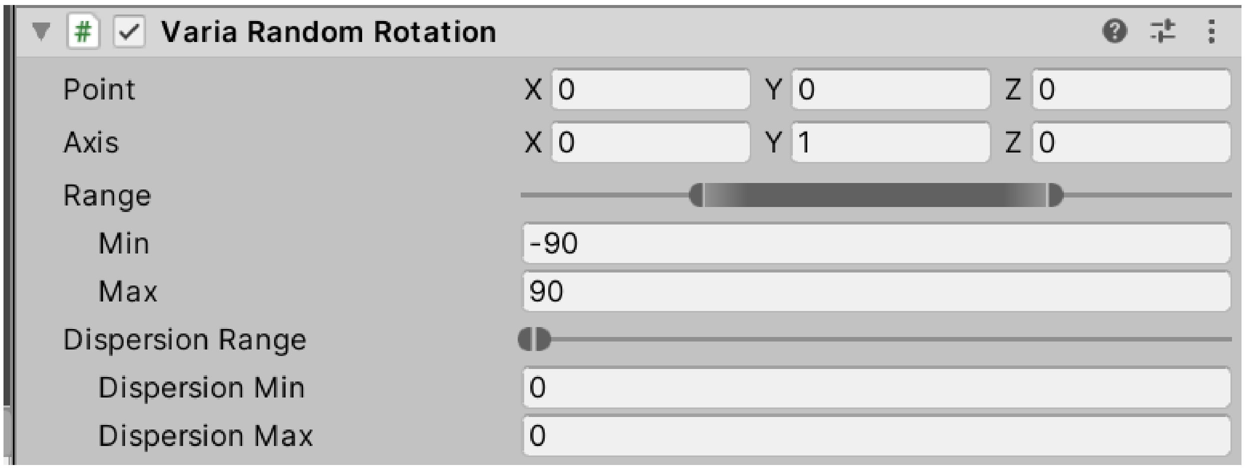
Varia Random Position

Relative To	Local
Min X	0
Max X	0
Min Y	0
Max Y	0
Min Z	0
Max Z	0

RandomPosition translates the object by a random vector. The vector is picked from inside a box specified by min/max x/y/z. The box is normally oriented the same way as the object (i.e. local space), but you can change the Relative To to World or Parent to use other co-ordinate systems.

When editing, a gizmo displays the box in the correct orientation.

Rotation



Varia Random Rotation

Point	X 0	Y 0	Z 0
Axis	X 0	Y 1	Z 0
Range			
Min	-90		
Max	90		
Dispersion Range			
Dispersion Min	0		
Dispersion Max	0		

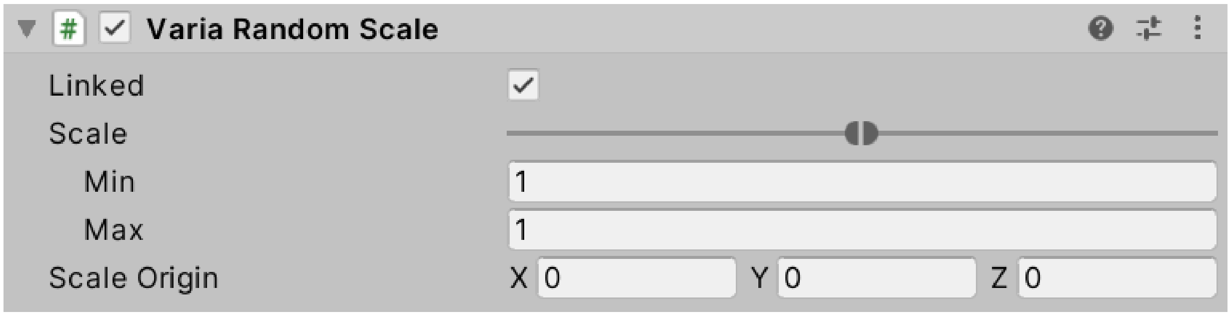
Random Rotation rotates an object a random number of degrees around a given local point and axis. So it works similarly to Transform.RotateAround.

When editing, a gizmo draws the axis of rotation, and an arc of the min/max number of degrees.

Additionally, you can set a random dispersion, which causes the object to randomly turn a number of degrees in a random axis perpendicular to the specified axis.

When editing, dispersion is shown as a pair of circles.

Scale


Varia Random Scale

Linked	✓
Scale	
Min	1
Max	1
Scale Origin	X 0 Y 0 Z 0

Random Scale randomly scales an object by a random amount. If linked is true, all three axes are scaled by the same amount, otherwise each is scaled independently.

If scale origin is set to a local, then the component also translates the transform, so that the given position doesn't move after scaling.

Varia Random Tint

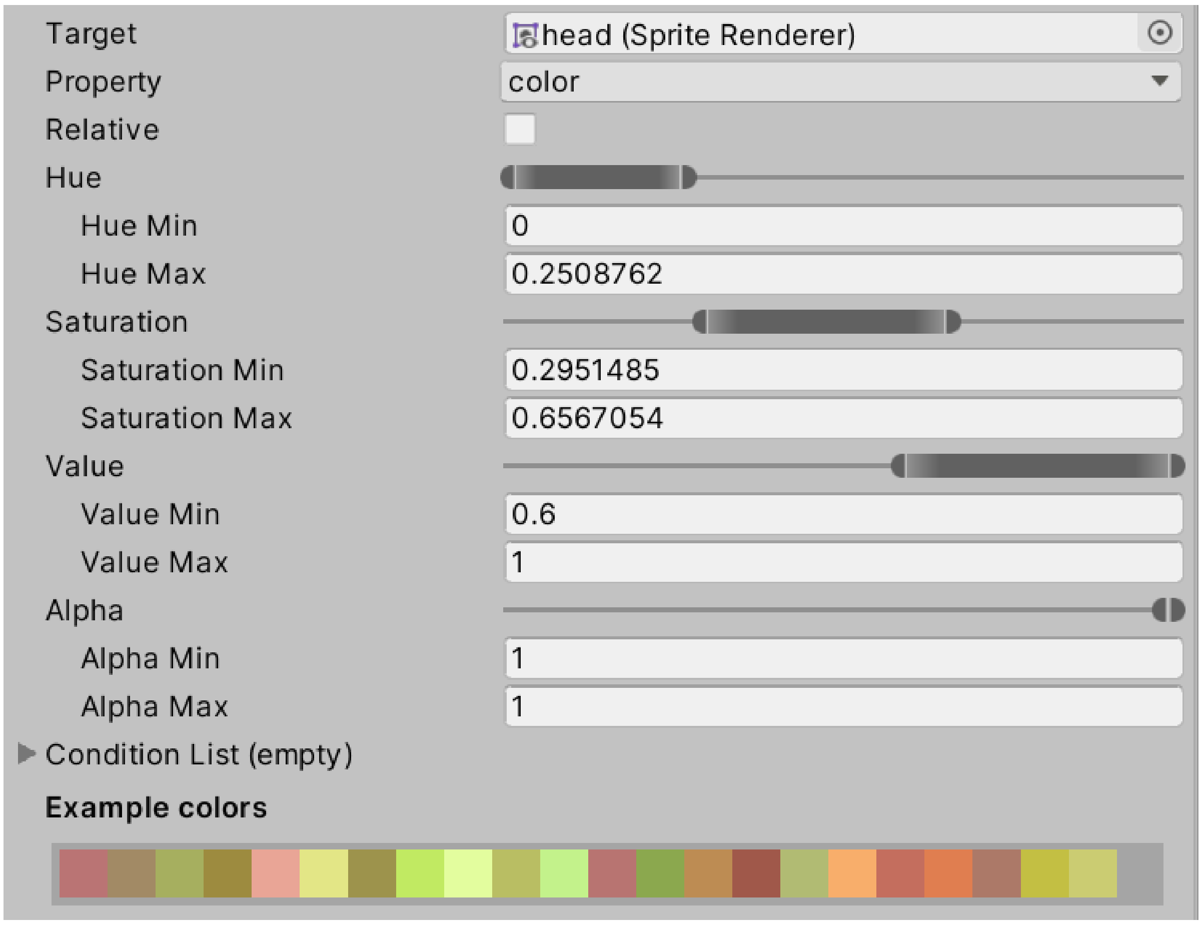
Random Tint

Random Tint randomly colorizes objects. It supports setting any component property, or a material property.

If changing a material, please note that Unity has two different ways to it:

- Varia can clone then change the Material itself. This is the most reliable, however it is not the best for performance and doesn't work outside of play mode.
- Varia can use the MaterialPropertyBlock API. This is more efficient, however, not all shaders support it.

You can select between these by chosing "Material" or "Material Property Block" in the property dropdown.



Target	head (Sprite Renderer)
Property	color
Relative	☐
Hue	
Hue Min	0
Hue Max	0.2508762
Saturation	
Saturation Min	0.2951485
Saturation Max	0.6567054
Value	
Value Min	0.6
Value Max	1
Alpha	
Alpha Min	1
Alpha Max	1

Condition List (empty)

Example colors

Settings

Target / Property

This will automatically be set to the color property of the Renderer component. But you can set any color based property (including materials), the same way as described in Varia Random Value.

Hue / Saturation / Value / Alpha

These four ranges determine the actual color to set, in the same fashion as Random.ColorHSV.

When `relative` is true, Hue, Saturation and Value are *added* to the base color to determine the new value. So the UI will permit negative values. Alpha is multiplied, so negative values are not useful.

Relative

If true, then a base color is loaded, and the randomized color is used to offset the base color. The base color usually comes from object itself, but it can come from a parent by setting **Relative Parent** to the number of steps upwards in the Unity hierarchy to look.

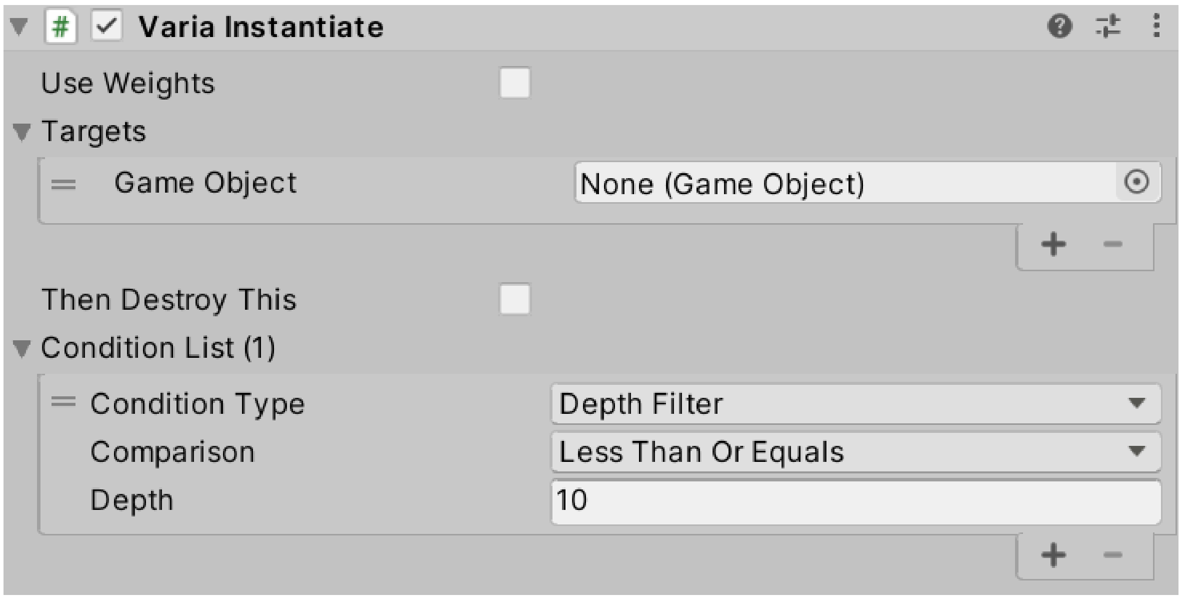
VariaInstantiate

The VariaInstantiate component picks a random prefab from a list, and instantiates it in the same position as the original component.

The instantiated objects will apply all their varia components too, including more instances of VariaInstantiate, so you can create quite deep nested structures.

▯ Note

You are recommended to only instantiate prefabs, or objects marked with VariaPrototype. Otherwise you run the risk of Varia components being applied twice - once on the original object, and a second time when you instantiate a copy of it.



Varia Instantiate

Use Weights ☐

Targets

Game Object: None (Game Object)

Then Destroy This ☐

Condition List (1)

Condition Type: Depth Filter

Comparison: Less Than Or Equals

Depth: 10

Depth and recursion

Every time a game object is instantiated inside of another one, a hidden variable called `depth` is increased to track how nested the current game object is. `depth` can be tested for using conditions.

You can even set up *recursive* instantiations, where a prefab A creates a prefab B, which creates prefab A again, which repeats until a condition on depth disables further instances. This technique is very powerful, and is described in more detail in the Recursion tutorial.

▯ Warning

Varia has difficulty dealing with direct recursion, i.e. when you set up VariaInstantiate to instantiate a parent of the object the component is on. This can cause an error in some cases. To fix it, either make a "wrapper" prefab that contains the prefab you want to instantiate, or ensure that the object is created with VariaUtils.Instantiate instead of Unity's GameObject.Instantiate.

Settings

Use Weights

By default, each value in the list is equally likely to be picked. If you turn on use weights, then you can set extra values to make some values more or less likely. Each value is picked with frequency proportional to the supplied weight.

Targets

The list of game objects to instantiate. You are recommended to only instantiate prefabs, or objects marked with VariaPrototype.

Then Destroy This

If true, destroys the game object the VariaInstantiate component is on. This can be used to make the instantiation work as a replacement instead.

Recursion

The VariaInstantiate component is a useful part of Varia. It causes an arbitrary prefab or other object to be instantiated when the original object is. You could randomly add a hat object to creatures in your game.

But if you start to play around with it, you'll notice it's a bit more subtle than some of the other components Varia offers. The instantiated object can itself have Varia components attached, which get run when it's created. If those components include another VariaInstantiate component, then it'll run and create a third thing. And that third thing might go on to do even more...

In fact, you might have a series of prefabs, each of which creates the next in the series, and the final one creates the first once, starting the cycle over and over forever. This is the essence of Recursion.

We can harness this sort of behaviour to create a whole class of generated objects with Varia that would otherwise be impossible. Obviously, creating objects forever is not really feasible, so we'll put in a depth limit that stops creation after a certain number of objects.

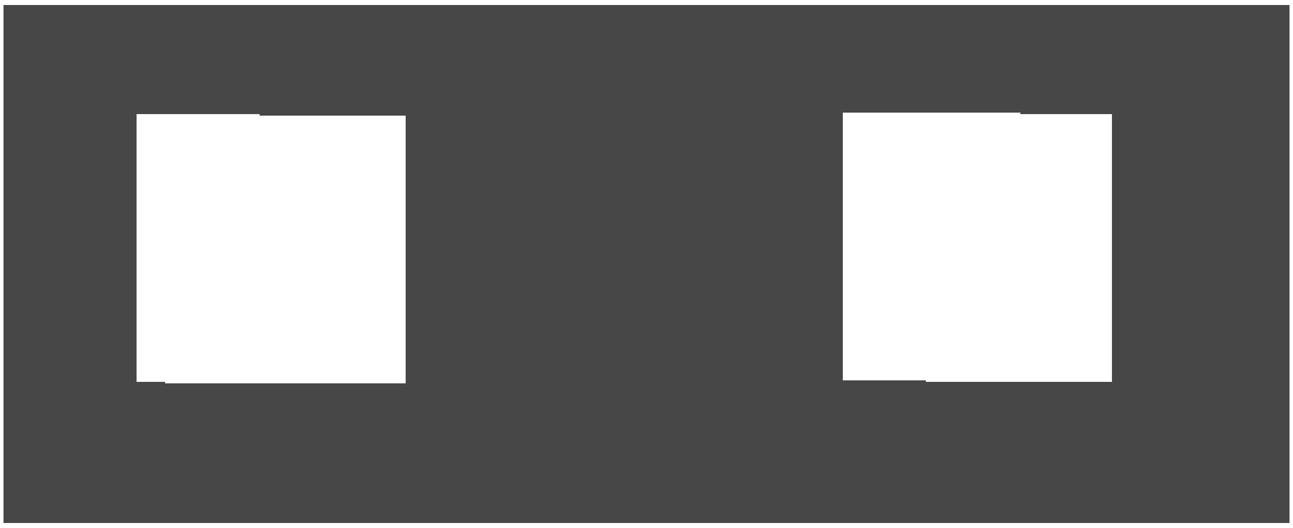
In this tutorial, we're going to recreate the Golden Spiral fractal found in the Fractals sample.

Initial setup

First, create a new sprite from the white_square asset in the Fractals sample. Call it `golden_spiral`, and give it the `VariaPrototype` component. Adding this component disables all other varia components on it. That means when we start the scene, it'll remain unchanged. That's important as we're going to copy `golden_spiral` many times, and need it to start from a consistent position.

Instead of adding `VariaPrototype`, you could just make it a prefab. However, it's handy to have the object in the scene itself so you can edit it without constantly changing scenes in the editor.

Next, we're going to set up a "Previewer". Previewers automatically instantiate a given object, so are very useful for viewing in the editor what would occur when an object is created. Create an empty, add the `VariaPreviewer` component, check "Continuous Refresh" and "Refresh in Editor", and set the target to `golden_spiral`. Now any changes you make to `golden_spiral` will be instantly reflected in the previewer. So you should see two white squares - one for `golden_spiral`, and one for the previewer.



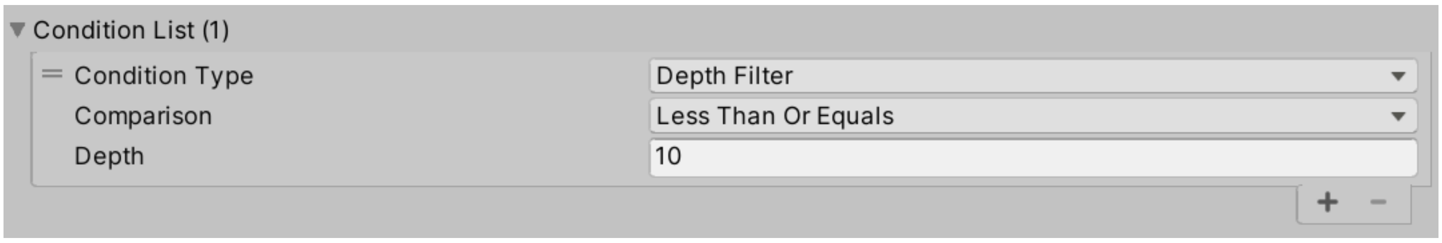
Adding the recursion

Create an empty object as a child of `golden_spiral`, and call it, `sub`. Set the scale of it to 0.7, hen add a `VariaInstantiate` component. Finally, set the Targets array of that component to reference `golden_spiral`.

Now move `sub` around a bit with Unity's Move Tool. You should immediately see that 9 additional squares are visible in the previewer, each smaller than the last. As you move sub, they react in an interesting way.

What is happening is that `sub` is creating a fresh copy of `golden_spiral` using `sub`'s transform information. So, as we scaled `sub` 0.7, the new copy of `golden_spiral` is also scaled by 0.7. That new copy has it's own `sub`, which is scaled by 0.7 twice, giving it a total size of 0.49, just under half the size of the original. It creates a `golden_spiral` with that size, and so on.

The process stops after 10 copies have been created. That is because, by default, `VariaInstantiate` has a limit of 10 repetitions:


Condition List (1)

Condition Type	Depth Filter
Comparison	Less Than Or Equals
Depth	10

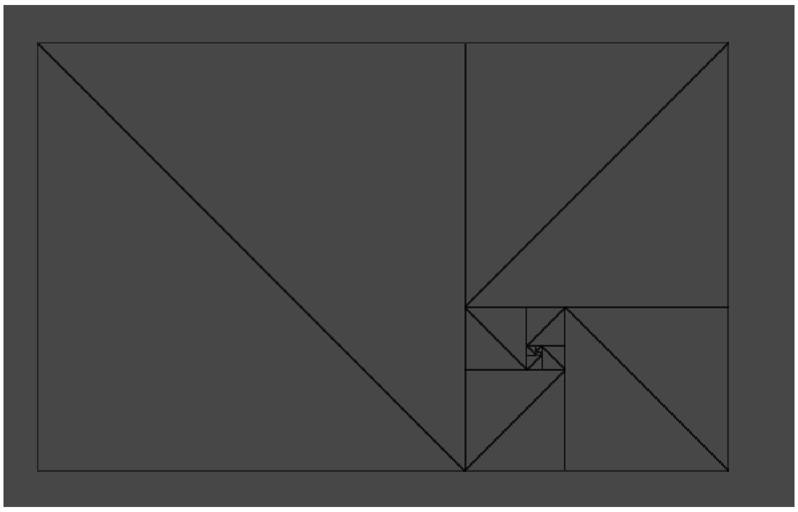
▯ Warning

Setting the depth filter to too high a value, or removing it, will cause Unity to create too many objects. That can crash the editor!

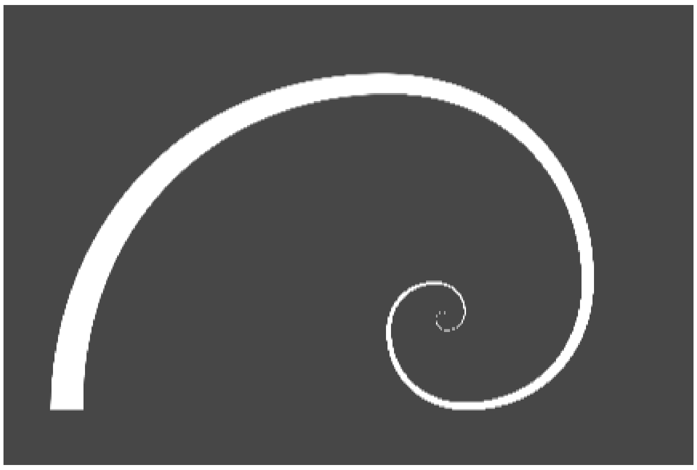
To recreate the shape of the golden spiral, `sub` should have the following settings:

```
Position: (1.618034,        0,        0)
Rotation: (0,               0,      270)
Scale:    (0.618034, 0.618034, 0.618034)
```

At these values, the rectangles all interlock in a tight spiral, as you can see from this wireframe view:



To finish things off, replace the use of white_square.png with golden_curve.png, which draws an appropriate arc for each sprite. You should be rewarded with the full spiral.



More complicated patterns can be done via careful use of what to instantiate. The fractals and tree samples demonstrate a few ideas.

For more information on recursion, see recursion. Sorry, a little programmer humour there...

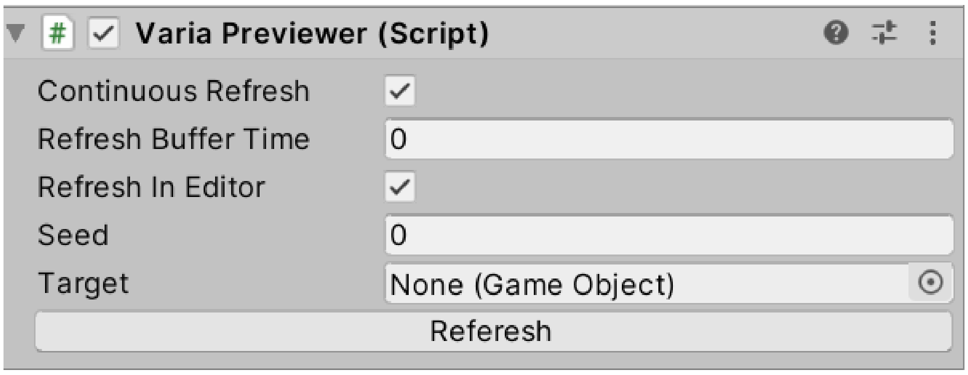
Previewer

VariaPreviewer is a component that automatically instantiates a given object or prefab.

It's useful for doing live previews in the editor as you configure your objects.

The recursion tutorial includes a bit of detail about how to set up and use the previewer.

Settings



Varia Previewer (Script)

Continuous Refresh	✓
Refresh Buffer Time	0
Refresh In Editor	✓
Seed	0
Target	None (Game Object)

Target

The unity object that should be instantiated. This should usually be a prefab, or it should have component `VariaPrototype`. Otherwise, that component will run any varia components it has *before* the previewer instantiates it and runs them a second time.

Refresh In Editor

If true, then the previewer runs in the editor when opening the scene.

Note that the objects created are marked as DontSave so they will not get saved in the scene.

Continous Refresh

If Refresh in Editor is set and this is true, then the previewer will run every time you change the scene.

Refresh Buffer Time

Forces continouous refresh to be a little less immediate, useful if you are suffering performance issues.

Seed

If non-zero, initializes Unity's random number generator. This can be used to make the preview repeateable, which is useful if you are seeing too much noise.

Conditions

Conditions are a way of of turning on/off Varia components automatically. Before applying a Varia component, all of the conditions of that component are evaluated, and if any of them are false, then that component will be skipped.

This is a great way to add systematic variation to your objects, and turns Varia into a mini-programming language.

▯ Note

Note a few components, such as VariaKeep treat conditions differently, rather than always skipping the component.

There's multiple conditions you can add to a component, described below.

Random

A Random condition simply randomly decides if the condition passes. You can set the random chance between 0 and 1, where 0 means never passes, and 1 means always passes.

This is particularly useful with components like VariaKeep / VariaDestroy, as it allows you to randomly choose whether to include an optional item.

Depth Filter

A Depth Filter condition checks the hidden `depth` property against a fixed value. Depth starts at zero and is increased by one for every nested use of VariaInstantiate.

This condition's main use is to control recursive behaviour. See the docs on Instantiate or the recursion tutorial.

Extending Varia

You can easily extend Varia by making more components that inherit from VariaBehaviour. Then override the Apply method to control what happens when your new behaviour is run. Conditions are checked beforehand, and the context supplied with details of the application.

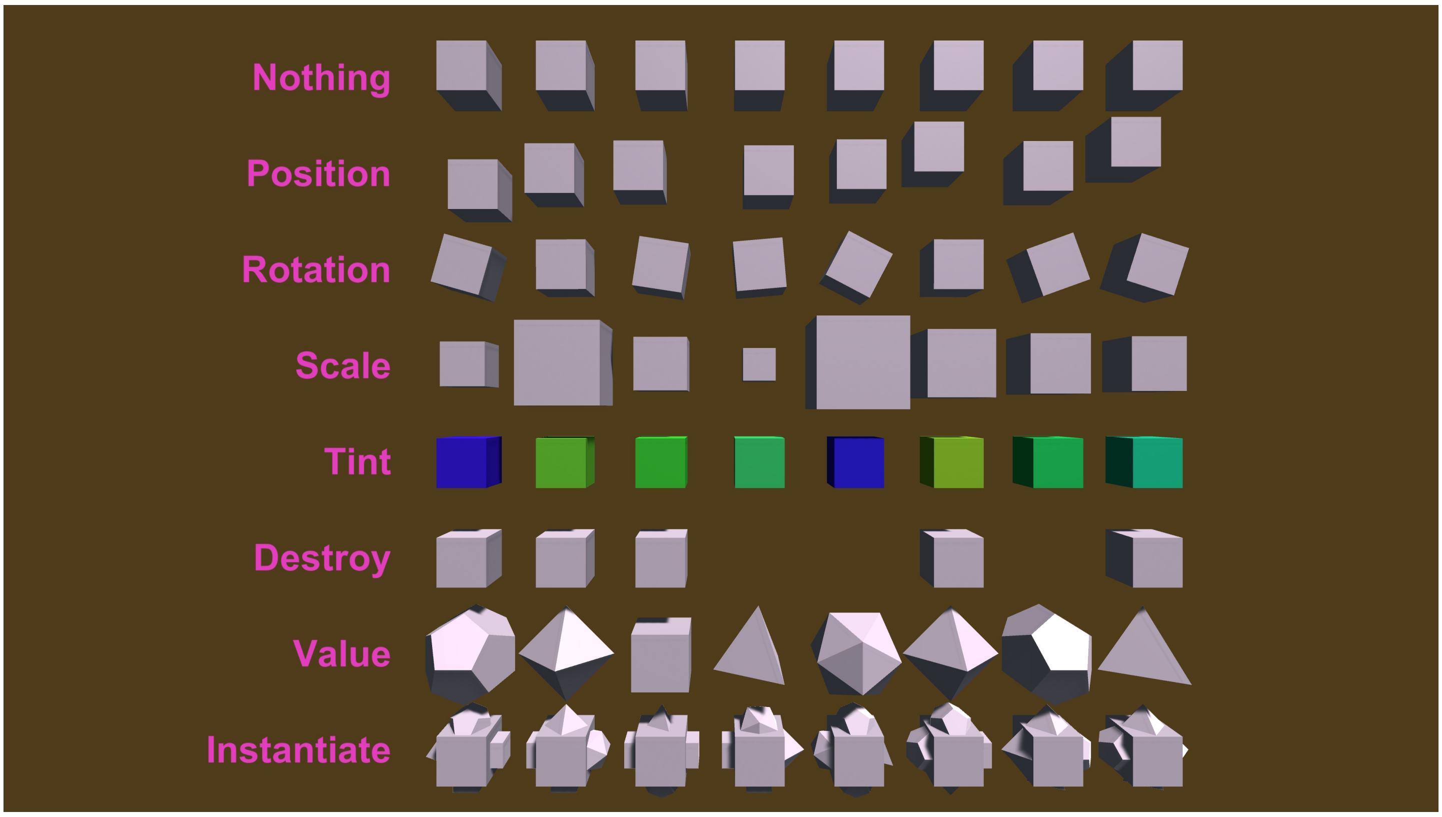
Varia's Target-Property system is available for re-use in your own code. See VariaReflection.

Samples

Here is full list of samples supplied with Varia.

01 - Overview

Demonstrates the usage of all varia behaviours, one per row. Each behaviour alters the 8 white cubes in some way.



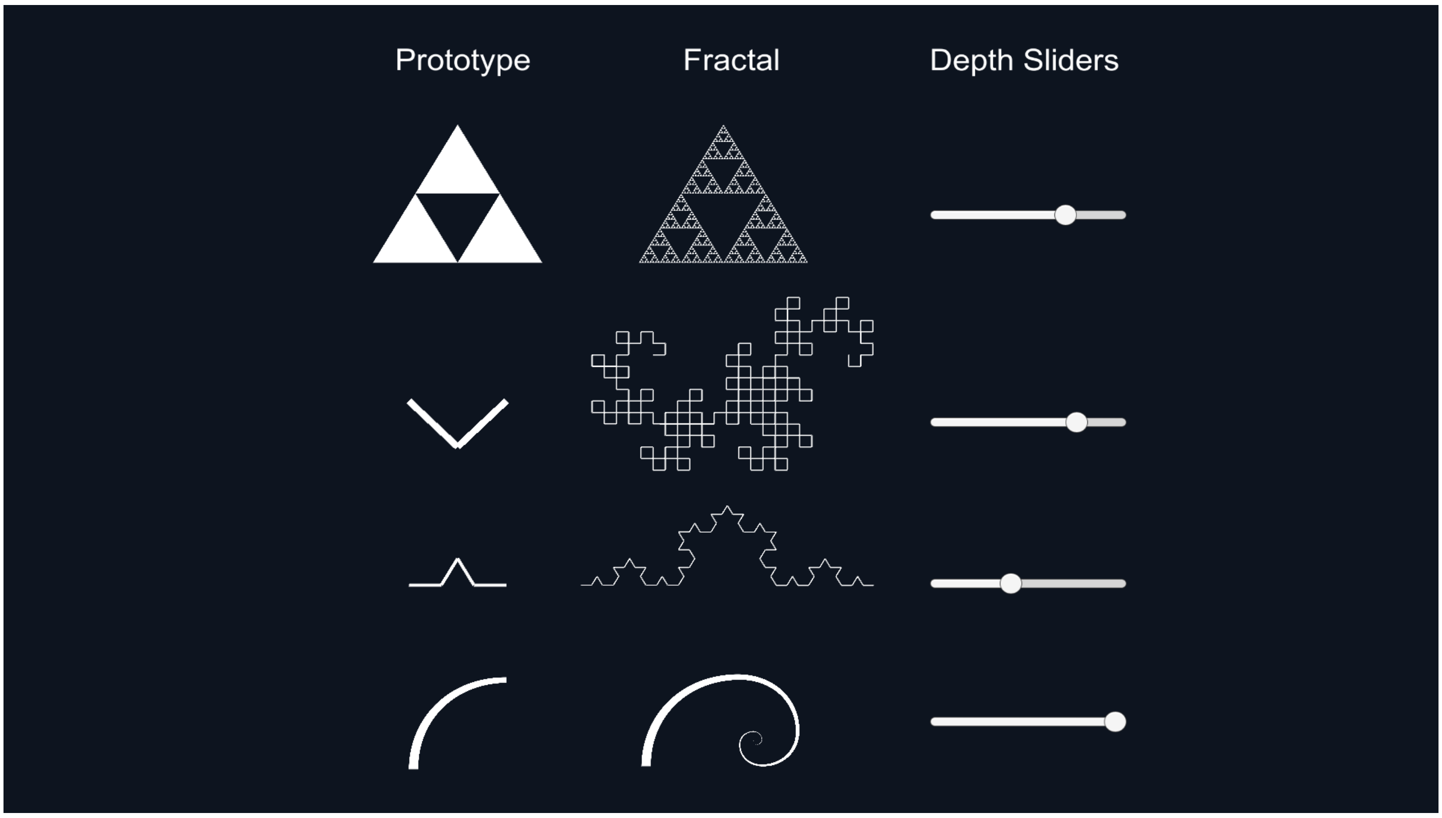
02 - Portrait Generator

Using assets from Noble Avatar (CC-BY 3.0), this show cases a random face generator.

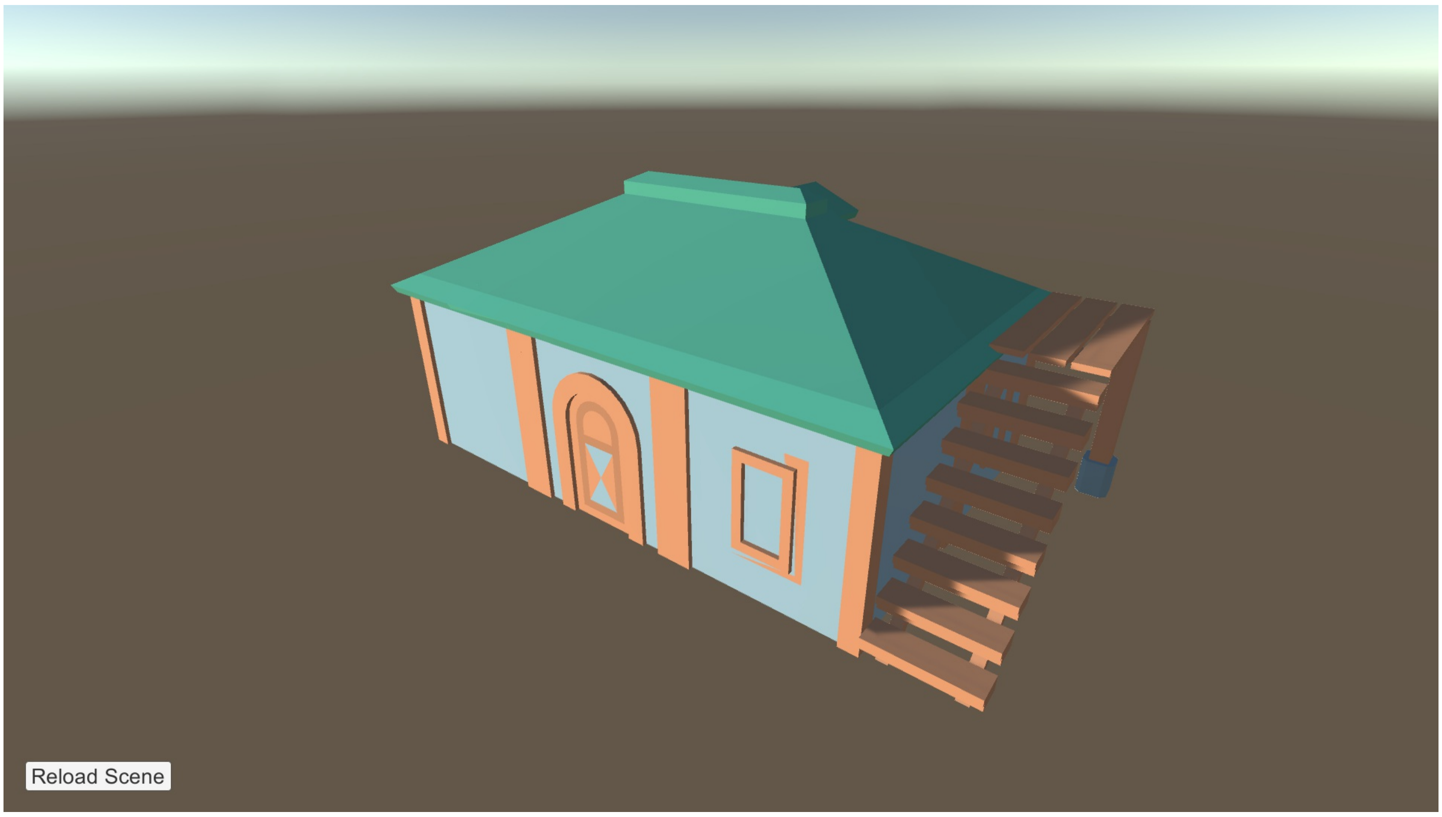
The sample makes heavy use of Random Value to swap between alternative images.

03 - Fractals

The Instantiate behaviour can be run recusively to generate these well known fractals. More details on Recursion

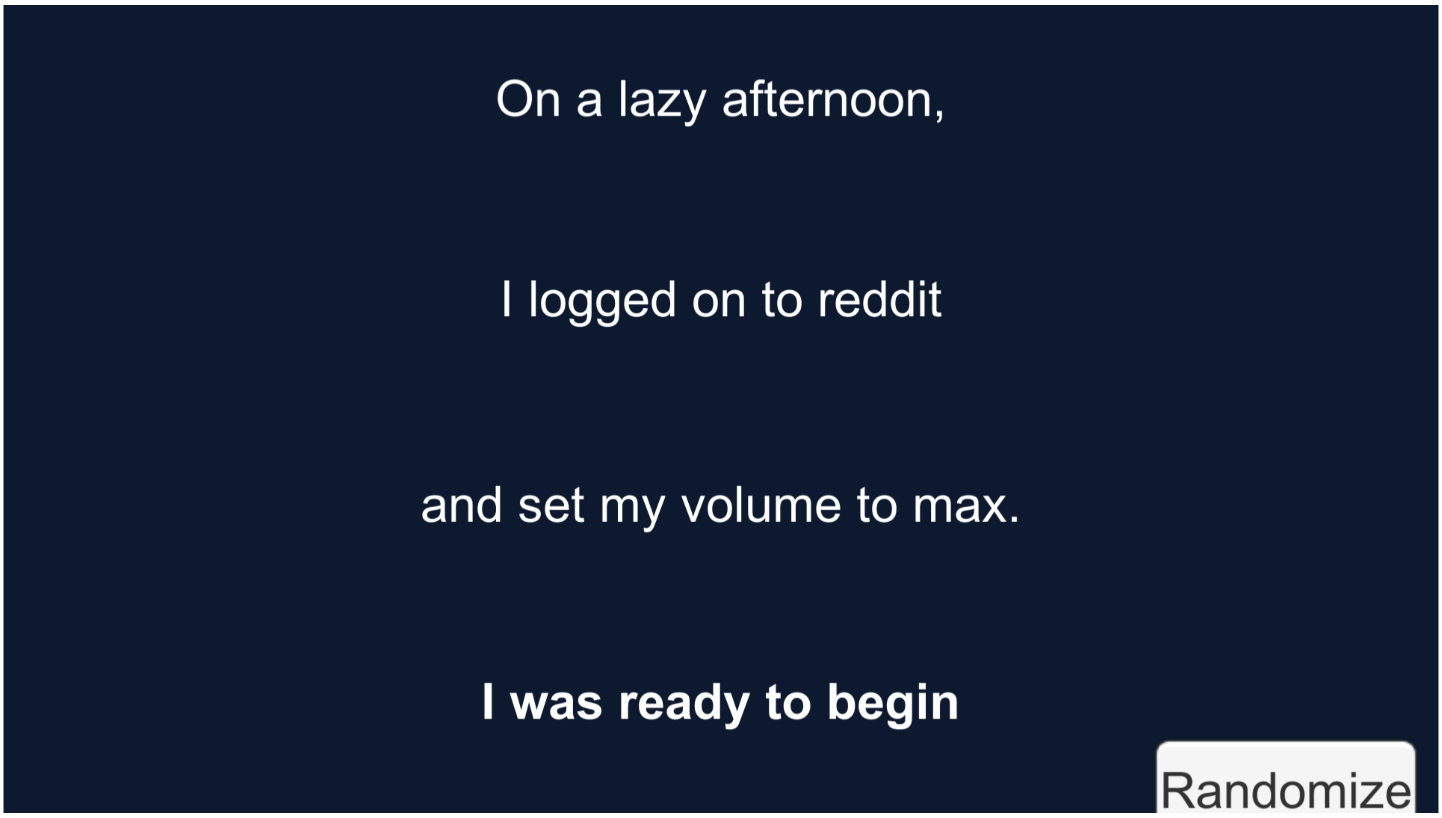


04 - FantasyHouse



05 - Random Text

Picks random text strings. Somehwat inspired by Spelunky's opening text.


On a lazy afternoon,

I logged on to reddit

and set my volume to max.

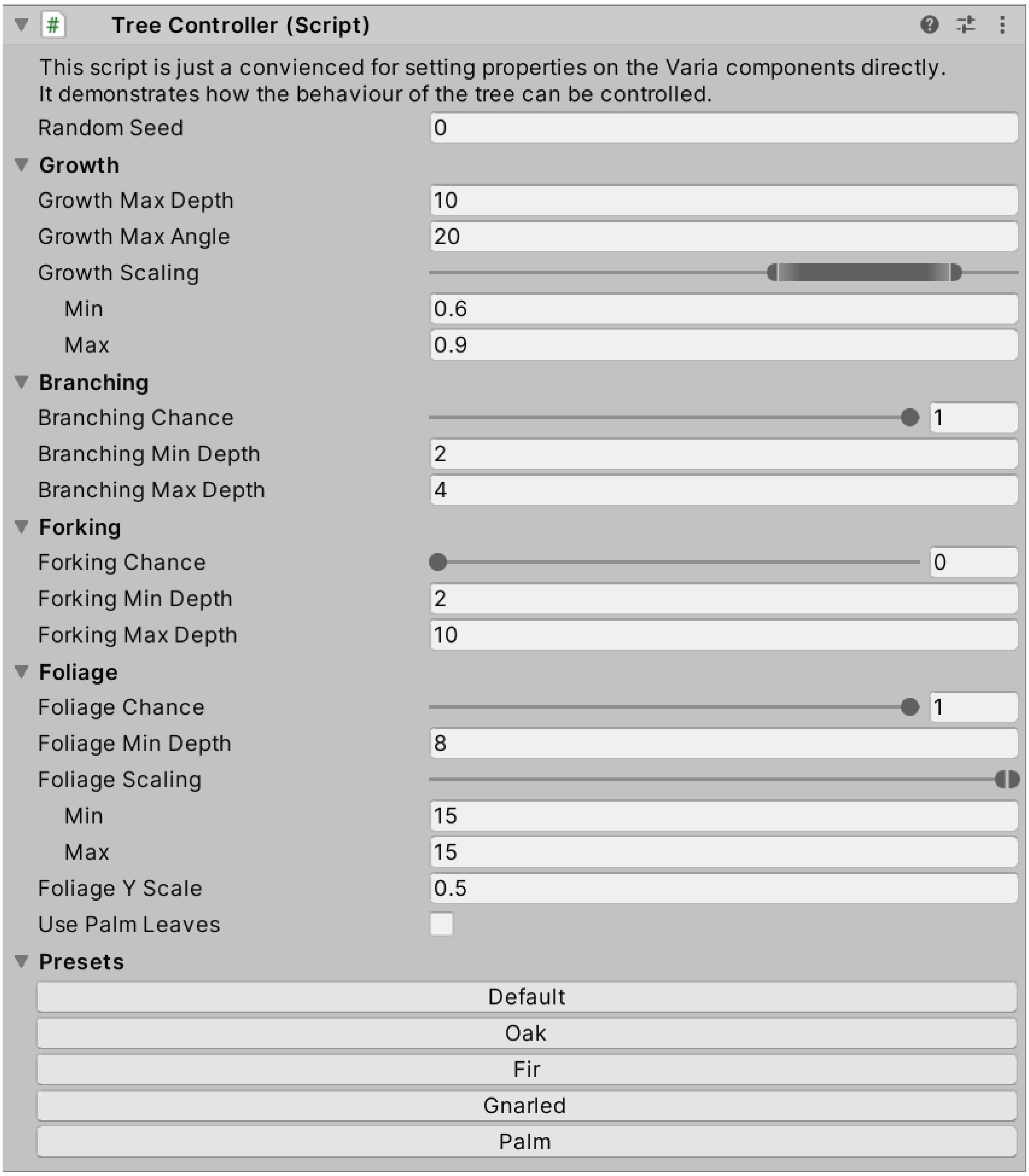
I was ready to begin

06 - Trees

An advanced example of recursion to generate tree. Allen Pike has an article on a similar idea.

The mesh used for the leaf foliage has non-standard normals to give a more diffuse look.

The Trees sample comes with a controller script that lets you edit all the pertinent parameters at once.

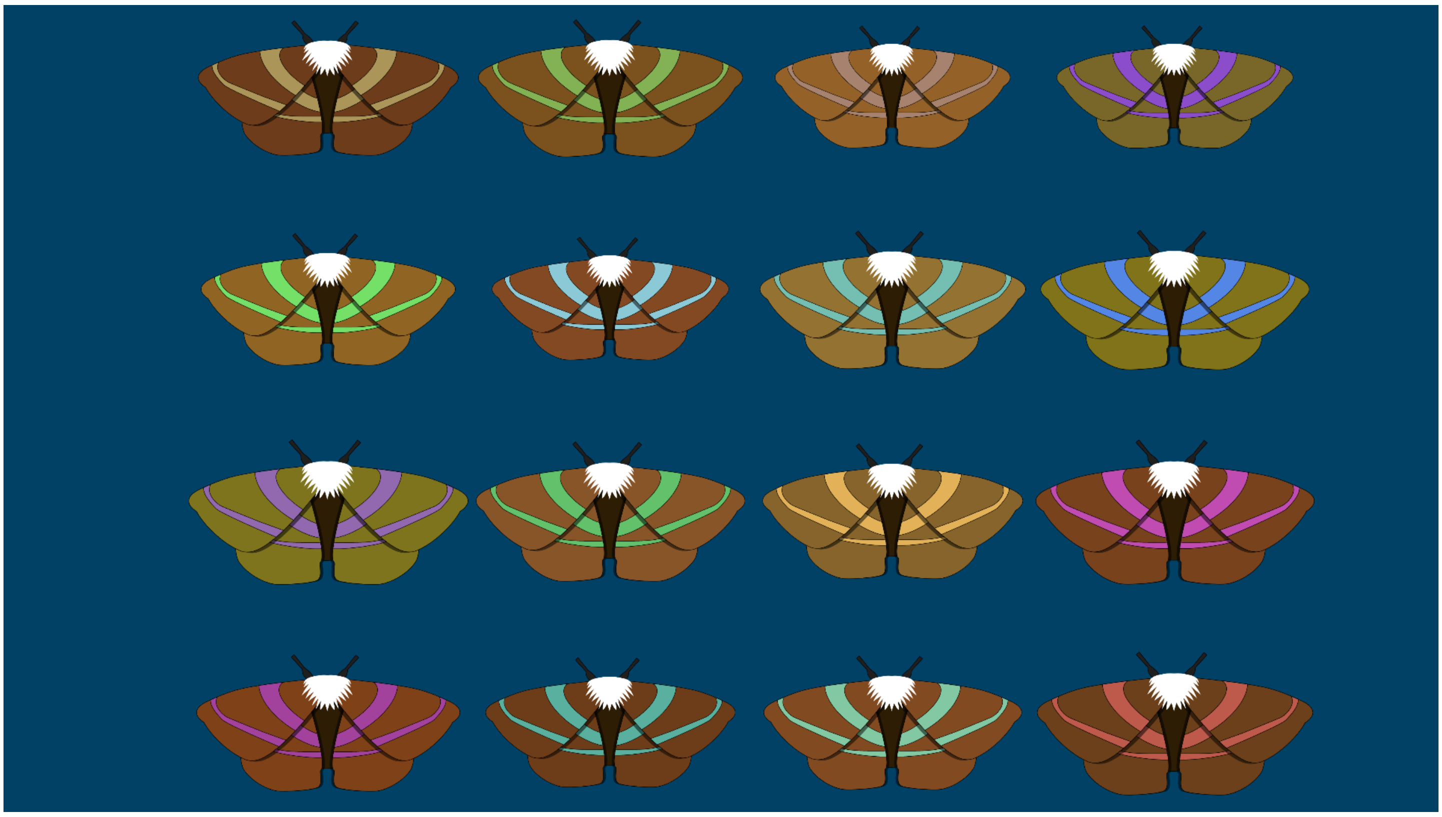

Tree Controller (Script)

This script is just a convienced for setting properties on the Varia components directly.
It demonstrates how the behaviour of the tree can be controlled.

Random Seed	0
Growth	
Growth Max Depth	10
Growth Max Angle	20
Growth Scaling	
Min	0.6
Max	0.9
Branching	
Branching Chance	1
Branching Min Depth	2
Branching Max Depth	4
Forking	
Forking Chance	0
Forking Min Depth	2
Forking Max Depth	10
Foliage	
Foliage Chance	1
Foliage Min Depth	8
Foliage Scaling	
Min	15
Max	15
Foliage Y Scale	0.5
Use Palm Leaves	☐

Presets

07 - Moths



Release notes

v1.1.0

- Add caveats about Ahead-of-time compilation
- Add "Freeze" menu item for making changes permanent.

v1.0.0

- Initial release

Namespace Varia

Classes

VariaBehaviour

Base class for all varia components that actually do something. Simply inherit, and override the Apply() method to make a new component.

VariaCondition

VariaConditionList

VariaContext

VariaDestroy

Destroys the game object. You should add conditions to this component or it is mostly useless.

VariaInstantiate

Picks a random prefab from a list, and instantiates it with VariaUtils.Instantiate. This may recursively instantiate more objects itself, which is tracked as the "depth". After the instantiation, this object is deleted.

Warning: There are some issues with using a target that is a direct parent of this component. To fix this, either make an extra prefab to avoid the issue, or use VariaUtils.Instantiate instead of normal instantiation.

VariaKeep

Destroys the GameObject if the conditions are *not* met. You should add conditions to this component or it is mostly useless.

VariaMirror

Abstraction for a a readable or writable property of a given instance.

VariaPreviewer

Utility for automatically calling VariaUtils.Instantiate. This is particularly useful in the editor to get a live preview of results.

VariaProperty

A cut down version of System.Reflection.PropertyInfo

VariaPrototype

Add this to any game objects with VariaBehaviours that you want to instantiate multiple times. It disables all VariaBehaviour on this object and children, so it is pristine for copying. It's not necessary for prefabs.

VariaRandomPosition

Offsets the position, randomly

VariaRandomRotation

Rotates the object randomly around a given axis.

Two sorts of rotation are supported:

- Rotating around the axis (rolling) using min and max.
- Rotating away from the axis (pitch / yaw) using dispersionMin and dispersionMax

VariaRandomScale

Changes transform.localScale randomly.

VariaRandomTint

Randomly sets the color of a MeshRenderer or SpriteRenderer component.

VariaRandomValue

Sets any property of any component to a value chosen randomly from a list. Only properties of types subclassing UnityEngine.Object are supported currently.

VariaReflection

VariaReflection is a simplified version of C# reflection, with an emphasis on reading and writing.

The main feature is given a expression, it lets you read and write values for the field corresponding to that expression. An expression is build as follows: expression ::= property_name | property_name "." expression | "propertyBlock" "." material_property_name "." type

The first form indicates a property of the target object itself. The second form evaluates the sub-expression on the value of named property of the target object. The third form corresponds to Renderer.SetPropertyBlock (https://docs.unity3d.com/ScriptReference/Renderer.SetPropertyBlock.html)

Material property block properties behave particularly strangely:

- They must have the type encoded in the name as it's not available at runtime.
- They are not listed when exploring the properties of an object (though you can use VariaMaterialPropertyBlockReflection to get them)

VariaUtils

VariaWeightedValue

WeightedGameObject

Enums

RelativeTo

VariaComparison

VariaConditionType

VariaSerializedValueType

Enum RelativeTo

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public enum RelativeTo
```

Fields

NAME	DESCRIPTION
Local	
Parent	
World	

Class VariaBehaviour

Base class for all varia components that actually do something. Simply inherit, and override the Apply() method to make a new component.

Inheritance

Object
VariaBehaviour
VariaDestroy
VariaInstantiate
VariaKeep
VariaRandomPosition
VariaRandomRotation
VariaRandomScale
VariaRandomTint
VariaRandomValue

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public abstract class VariaBehaviour : MonoBehaviour
```

Fields

conditionList

Declaration

```
public VariaConditionList conditionList
```

Field Value

TYPE	DESCRIPTION
VariaConditionList	

Methods

Apply(VariaContext)

Override this to control what happens when all the conditions are met

Declaration

```
public virtual void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

NoApply(VariaContext)

Override this to control what happens when a condition is missed

Declaration

```
public virtual void NoApply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

OnEnable()

Declaration

```
protected void OnEnable()
```

Enum VariaComparison

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public enum VariaComparison
```

Fields

NAME	DESCRIPTION
Equals	
GreaterThan	
GreaterThanOrEquals	
LessThan	
LessThanOrEquals	
NotEquals	

Class VariaCondition

Inheritance

Object

VariaCondition

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaCondition
```

Fields

comparison

Declaration

```
public VariaComparison comparison
```

Field Value

TYPE	DESCRIPTION
VariaComparison	

conditionType

Declaration

```
public VariaConditionType conditionType
```

Field Value

TYPE	DESCRIPTION
VariaConditionType	

depth

Declaration

```
public int depth
```

Field Value

TYPE	DESCRIPTION
Int32	

randomChance

Declaration

```
public float randomChance
```

Field Value

TYPE	DESCRIPTION
Single	

Class VariaConditionList

Inheritance

Object

VariaConditionList

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaConditionList
```

Fields

conditions

Declaration

```
public List<VariaCondition> conditions
```

Field Value

TYPE	DESCRIPTION
List<VariaCondition>	

Enum VariaConditionType

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public enum VariaConditionType
```

Fields

NAME	DESCRIPTION
DepthFilter	
Random	

Class VariaContext

Inheritance

Object

VariaContext

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaContext
```

Constructors

VariaContext()

Declaration

```
public VariaContext()
```

Fields

randomState

Declaration

```
public Random.State randomState
```

Field Value

TYPE	DESCRIPTION
Random.State	

Properties

current

Declaration

```
public static VariaContext current { get; }
```

Property Value

TYPE	DESCRIPTION
VariaContext	

depth

Declaration

```
public int depth { get; }
```

Property Value

TYPE	DESCRIPTION
Int32	

log

Declaration

```
public bool log { get; set; }
```

Property Value

TYPE	DESCRIPTION
Boolean	

recordUndo

Declaration

```
public bool recordUndo { get; set; }
```

Property Value

TYPE	DESCRIPTION
Boolean	

Methods

Freeze(GameObject)

Declaration

```
public void Freeze(GameObject go)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	go	

Instantiate(GameObject)

Declaration

```
public GameObject Instantiate(GameObject original)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Transform)

Declaration

```
public GameObject Instantiate(GameObject original, Transform parent)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Transform	parent	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Transform, Boolean)

Declaration

```
public GameObject Instantiate(GameObject original, Transform parent, bool worldPositionStays)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Transform	parent	
Boolean	worldPositionStays	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Vector3, Quaternion)

Declaration

```
public GameObject Instantiate(GameObject original, Vector3 position, Quaternion rotation)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Vector3	position	
Quaternion	rotation	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Vector3, Quaternion, Transform)

Declaration

```
public GameObject Instantiate(GameObject original, Vector3 position, Quaternion rotation, Transform parent)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Vector3	position	
Quaternion	rotation	
Transform	parent	

Returns

TYPE	DESCRIPTION
GameObject	

Class VariaDestroy

Destroys the game object. You should add conditions to this component or it is mostly useless.

Inheritance

Object
VariaBehaviour
VariaDestroy

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaDestroy : VariaBehaviour
```

Constructors

VariaDestroy()

Declaration

```
public VariaDestroy()
```

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

Class VariaInstantiate

Picks a random prefab from a list, and instantiates it with VariaUtils.Instantiate. This may recursively instantiate more objects itself, which is tracked as the "depth". After the instantiation, this object is deleted.

Warning: There are some issues with using a target that is a direct parent of this component. To fix this, either make an extra prefab to avoid the issue, or use VariaUtils.Instantiate instead of normal instantiation.

Inheritance

Object
VariaBehaviour
VariaInstantiate

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaInstantiate : VariaBehaviour
```

Constructors

VariaInstantiate()

Declaration

```
public VariaInstantiate()
```

Fields

targets

The list of game objects to instantiate, and their weights. You are recommended to only instantiate prefabs, or objects marked with VariaPrototype

Declaration

```
public List<WeightedGameObject> targets
```

Field Value

TYPE	DESCRIPTION
List<WeightedGameObject>	

thenDestroyThis

If true, destroys the game object the VariaInstantiate component is on. This can be used to make the instantiation work as a replacement instead.

Declaration

```
public bool thenDestroyThis
```

Field Value

TYPE	DESCRIPTION
Boolean	

useWeights

If enabled, the weight property alters the probabiliyt of picking that target. Otherwise, they are picked uniformly.

Declaration

```
public bool useWeights
```

Field Value

TYPE	DESCRIPTION
Boolean	

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

Class VariaKeep

Destroys the GameObject if the conditions are *not* met. You should add conditions to this component or it is mostly useless.

Inheritance

Object
VariaBehaviour
VariaKeep

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaKeep : VariaBehaviour
```

Constructors

VariaKeep()

Declaration

```
public VariaKeep()
```

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

NoApply(VariaContext)

Declaration

```
public override void NoApply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.NoApply(VariaContext)

Class VariaMirror

Abstraction for a a readable or writable property of a given instance.

Inheritance

Object

VariaMirror

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaMirror
```

Fields

getValue

Declaration

```
public Func<object, object> getValue
```

Field Value

TYPE	DESCRIPTION
Func<Object, Object>	

propertyType

Declaration

```
public Type propertyType
```

Field Value

TYPE	DESCRIPTION
Type	

setValue

Declaration

```
public Action<object, object> setValue
```

Field Value

TYPE	DESCRIPTION
Action<Object, Object>	

Class VariaPreviewer

Utility for automatically calling VariaUtils.Instantiate. This is particularly useful in the editor to get a live preview of results.

Inheritance

Object

VariaPreviewer

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaPreviewer : MonoBehaviour
```

Fields

continuousRefresh

Declaration

```
public bool continuousRefresh
```

Field Value

TYPE	DESCRIPTION
Boolean	

refreshBufferTime

Declaration

```
public float refreshBufferTime
```

Field Value

TYPE	DESCRIPTION
Single	

refreshInEditor

Declaration

```
public bool refreshInEditor
```

Field Value

TYPE	DESCRIPTION
Boolean	

seed

Declaration

```
public int seed
```

Field Value

TYPE	DESCRIPTION
Int32	

target

Declaration

```
public GameObject target
```

Field Value

TYPE	DESCRIPTION
GameObject	

Methods

Refresh()

Declaration

```
public void Refresh()
```

Class VariaProperty

A cut down version of System.Reflection.PropertyInfo

Inheritance

Object

VariaProperty

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaProperty
```

Fields

canRead

Declaration

```
public bool canRead
```

Field Value

TYPE	DESCRIPTION
Boolean	

canWrite

Declaration

```
public bool canWrite
```

Field Value

TYPE	DESCRIPTION
Boolean	

expression

Declaration

```
public string expression
```

Field Value

TYPE	DESCRIPTION
String	

name

Declaration

```
public string name
```

Field Value

TYPE	DESCRIPTION
String	

propertyType

Declaration

```
public Type propertyType
```

Field Value

TYPE	DESCRIPTION
Type	

Class VariaPrototype

Add this to any game objects with VariaBehaviours that you want to instantiate multiple times. It disables all VariaBehaviour on this object and children, so it is pristine for copying. It's not necessary for prefabs.

Inheritance

Object

VariaPrototype

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaPrototype : MonoBehaviour
```

Class VariaRandomPosition

Offsets the position, randomly

Inheritance

Object
VariaBehaviour
VariaRandomPosition

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaRandomPosition : VariaBehaviour
```

Fields

maxX

Declaration

```
public float maxX
```

Field Value

TYPE	DESCRIPTION
Single	

maxY

Declaration

```
public float maxY
```

Field Value

TYPE	DESCRIPTION
Single	

maxZ

Declaration

```
public float maxZ
```

Field Value

TYPE	DESCRIPTION
Single	

minX

Declaration

```
public float minX
```

Field Value

TYPE	DESCRIPTION
Single	

minY

Declaration

```
public float minY
```

Field Value

TYPE	DESCRIPTION
Single	

minZ

Declaration

```
public float minZ
```

Field Value

TYPE	DESCRIPTION
Single	

relativeTo

Inidcates what space the offset should be performed in.

Declaration

```
public RelativeTo relativeTo
```

Field Value

TYPE	DESCRIPTION
RelativeTo	

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

Class VariaRandomRotation

Rotates the object randomly around a given axis.

Two sorts of rotation are supported:

- Rotating around the axis (rolling) using min and max.
- Rotating away from the axis (pitch / yaw) using dispersionMin and dispersionMax

Inheritance

Object
VariaBehaviour
VariaRandomRotation

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaRandomRotation : VariaBehaviour
```

Fields

axis

Local axis of rotations

Declaration

```
public Vector3 axis
```

Field Value

TYPE	DESCRIPTION
Vector3	

dispersionMax

Max amount to rotate away from the axis.

Declaration

```
public float dispersionMax
```

Field Value

TYPE	DESCRIPTION
Single	

dispersionMin

Min amount to rotate away from the axis.

Declaration

```
public float dispersionMin
```

Field Value

TYPE	DESCRIPTION
Single	

max

Max amount to rotate around the axis

Declaration

```
public float max
```

Field Value

TYPE	DESCRIPTION
Single	

min

Min amount to rotate around the axis

Declaration

```
public float min
```

Field Value

TYPE	DESCRIPTION
Single	

point

Point to keep fixed during rotation

Declaration

```
public Vector3 point
```

Field Value

TYPE	DESCRIPTION
Vector3	

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

Class VariaRandomScale

Changes transform.localScale randomly.

Inheritance

Object
VariaBehaviour
VariaRandomScale

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaRandomScale : VariaBehaviour
```

Fields

linked

If true, X,Y and Z are all scaled together, otherwise they are independently scaled.

Declaration

```
public bool linked
```

Field Value

TYPE	DESCRIPTION
Boolean	

maxX

Declaration

```
public float maxX
```

Field Value

TYPE	DESCRIPTION
Single	

maxY

Declaration

```
public float maxY
```

Field Value

TYPE	DESCRIPTION
Single	

maxZ

Declaration

```
public float maxZ
```

Field Value

TYPE	DESCRIPTION
Single	

minX

Declaration

```
public float minX
```

Field Value

TYPE	DESCRIPTION
Single	

minY

Declaration

```
public float minY
```

Field Value

TYPE	DESCRIPTION
Single	

minZ

Declaration

```
public float minZ
```

Field Value

TYPE	DESCRIPTION
Single	

scaleOrigin

The local point that should stay fixed while scaling

Declaration

```
public Vector3 scaleOrigin
```

Field Value

TYPE	DESCRIPTION
Vector3	

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

Class VariaRandomTint

Randomly sets the color of a MeshRenderer or SpriteRenderer component.

Inheritance

Object
VariaBehaviour
VariaRandomTint

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaRandomTint : VariaBehaviour
```

Fields

alphaMax

Declaration

```
public float alphaMax
```

Field Value

TYPE	DESCRIPTION
Single	

alphaMin

Declaration

```
public float alphaMin
```

Field Value

TYPE	DESCRIPTION
Single	

hueMax

Declaration

```
public float hueMax
```

Field Value

TYPE	DESCRIPTION
Single	

hueMin

Declaration

```
public float hueMin
```

Field Value

TYPE	DESCRIPTION
Single	

property

The name of the property on the target component.

Declaration

```
public string property
```

Field Value

TYPE	DESCRIPTION
String	

relative

Declaration

```
public bool relative
```

Field Value

TYPE	DESCRIPTION
Boolean	

relativeParent

Declaration

```
public int relativeParent
```

Field Value

TYPE	DESCRIPTION
Int32	

saturationMax

Declaration

```
public float saturationMax
```

Field Value

TYPE	DESCRIPTION
Single	

saturationMin

Declaration

```
public float saturationMin
```

Field Value

TYPE	DESCRIPTION
Single	

target

Specifices the specific component to set the value on.

Declaration

```
public Component target
```

Field Value

TYPE	DESCRIPTION
Component	

valueMax

Declaration

```
public float valueMax
```

Field Value

TYPE	DESCRIPTION
Single	

valueMin

Declaration

```
public float valueMin
```

Field Value

TYPE	DESCRIPTION
Single	

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

GetBaseColor()

Declaration

```
public Color? GetBaseColor()
```

Returns

TYPE	DESCRIPTION
Nullable<Color>	

GetColor(Boolean)

Declaration

```
public Color? GetColor(bool force = false)
```

Parameters

TYPE	NAME	DESCRIPTION
Boolean	force	

Returns

TYPE	DESCRIPTION
Nullable<Color>	

GetRelativeTarget()

Declaration

```
public Object GetRelativeTarget()
```

Returns

TYPE	DESCRIPTION
Object	

Class VariaRandomValue

Sets any property of any component to a value chosen randomly from a list. Only properties of types subclassing UnityEngine.Object are supported currently.

Inheritance

Object
VariaBehaviour
VariaRandomValue

Inherited Members

VariaBehaviour.conditionList
VariaBehaviour.OnEnable()
VariaBehaviour.NoApply(VariaContext)

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class VariaRandomValue : VariaBehaviour
```

Fields

property

The name of the property on the target component.

Declaration

```
public string property
```

Field Value

TYPE	DESCRIPTION
String	

target

Specifices the specific component to set the value on.

Declaration

```
public Component target
```

Field Value

TYPE	DESCRIPTION
Component	

useWeights

If true, the random choice from values is weighted, otherwise they are chosen uniformly.

Declaration

```
public bool useWeights
```

Field Value

TYPE	DESCRIPTION
Boolean	

values

The list of values to randomly choose from

Declaration

```
public List<VariaWeightedValue> values
```

Field Value

TYPE	DESCRIPTION
List<VariaWeightedValue>	

Properties

Mirror

Declaration

```
public VariaMirror Mirror { get; }
```

Property Value

TYPE	DESCRIPTION
VariaMirror	

Methods

Apply(VariaContext)

Declaration

```
public override void Apply(VariaContext context)
```

Parameters

TYPE	NAME	DESCRIPTION
VariaContext	context	

Overrides

VariaBehaviour.Apply(VariaContext)

Class VariaReflection

VariaReflection is a simplified version of C# reflection, with an emphasis on reading and writing.

The main feature is given a expression, it lets you read and write values for the field corresponding to that expression. An expression is build as follows: expression ::= property_name | property_name "." expression | "propertyBlock" "." material_property_name "." type

The first form indicates a property of the target object itself. The second form evaluates the sub-expression on the value of named property of the target object. The third form corresponds to Renderer.SetPropertyBlock (https://docs.unity3d.com/ScriptReference/Renderer.SetPropertyBlock.html)

Material property block properties behave particularly strangely:

- They must have the type encoded in the name as it's not available at runtime.
- They are not listed when exploring the properties of an object (though you can use VariaMaterialPropertyBlockReflection to get them)

Inheritance

Object

VariaReflection

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public static class VariaReflection
```

Methods

EvalExpression(Type, String)

Declaration

```
public static VariaMirror EvalExpression(Type targetType, string expression)
```

Parameters

TYPE	NAME	DESCRIPTION
Type	targetType	
String	expression	

Returns

TYPE	DESCRIPTION
VariaMirror	

EvalExpressionOrThrow(Type, String)

Declaration

```
public static VariaMirror EvalExpressionOrThrow(Type targetType, string expression)
```

Parameters

TYPE	NAME	DESCRIPTION
Type	targetType	
String	expression	

Returns

TYPE	DESCRIPTION
VariaMirror	

GetProperties(Type)

Declaration

```
public static List<VariaProperty> GetProperties(Type targetType)
```

Parameters

TYPE	NAME	DESCRIPTION
Type	targetType	

Returns

TYPE	DESCRIPTION
List<VariaProperty>	

GetValue(Object, String)

Declaration

```
public static object GetValue(object o, string expression)
```

Parameters

TYPE	NAME	DESCRIPTION
Object	o	
String	expression	

Returns

TYPE	DESCRIPTION
Object	

SetValue(Object, String, Object)

Declaration

```
public static void SetValue(object o, string expression, object value)
```

Parameters

TYPE	NAME	DESCRIPTION
Object	o	
String	expression	
Object	value	

Enum VariaSerializedValueType

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public enum VariaSerializedValueType
```

Fields

NAME	DESCRIPTION
AnimationCurve	
ArraySize	
Boolean	
Bounds	
BoundsInt	
Character	
Color	
Enum	
ExposedReference	
FixedBufferSize	
Float	
Generic	
Gradient	
Integer	
LayerMask	
ManagedReference	
ObjectReference	
Quaternion	
Rect	
RectInt	
String	

NAME	DESCRIPTION
Vector2	
Vector2Int	
Vector3	
Vector3Int	
Vector4	

Class VariaUtils

Inheritance

Object

VariaUtils

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public static class VariaUtils
```

Methods

GetNamePath(GameObject)

Gives the name of the all the ancestors of the current game object

Declaration

```
public static string GetNamePath(this GameObject go)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	go	

Returns

TYPE	DESCRIPTION
String	

Instantiate(GameObject)

Same behaviour as GameObject.Instantiate

Declaration

```
public static GameObject Instantiate(GameObject original)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Transform)

Same behaviour as GameObject.Instantiate

Declaration

```
public static GameObject Instantiate(GameObject original, Transform parent)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Transform	parent	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Transform, Boolean)

Same behaviour as GameObject.Instantiate

Declaration

```
public static GameObject Instantiate(GameObject original, Transform parent, bool worldPositionStays)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Transform	parent	
Boolean	worldPositionStays	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Vector3, Quaternion)

Same behaviour as GameObject.Instantiate

Declaration

```
public static GameObject Instantiate(GameObject original, Vector3 position, Quaternion rotation)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Vector3	position	
Quaternion	rotation	

Returns

TYPE	DESCRIPTION
GameObject	

Instantiate(GameObject, Vector3, Quaternion, Transform)

Same behaviour as GameObject.Instantiate

Declaration

```
public static GameObject Instantiate(GameObject original, Vector3 position, Quaternion rotation, Transform
parent)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	original	
Vector3	position	
Quaternion	rotation	
Transform	parent	

Returns

TYPE	DESCRIPTION
GameObject	

Class VariaWeightedValue

Inheritance

Object

VariaWeightedValue

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
[Serializable]
public class VariaWeightedValue : ISerializationCallbackReceiver
```

Fields

value

Declaration

```
[NonSerialized]
public object value
```

Field Value

TYPE	DESCRIPTION
Object	

weight

Declaration

```
public float weight
```

Field Value

TYPE	DESCRIPTION
Single	

Methods

CanSerialize(Type)

Declaration

```
public static bool CanSerialize(Type type)
```

Parameters

TYPE	NAME	DESCRIPTION
Type	type	

Returns

TYPE	DESCRIPTION
Boolean	

GetDefault(Type)

Declaration

```
public static object GetDefault(Type t)
```

Parameters

TYPE	NAME	DESCRIPTION
Type	t	

Returns

TYPE	DESCRIPTION
Object	

OnAfterDeserialize()

Declaration

```
public void OnAfterDeserialize()
```

OnBeforeSerialize()

Declaration

```
public void OnBeforeSerialize()
```

Class WeightedGameObject

Inheritance

Object

WeightedGameObject

Namespace: Varia

Assembly: cs.temp.dll.dll

Syntax

```
public class WeightedGameObject
```

Fields

gameObject

Declaration

```
public GameObject gameObject
```

Field Value

TYPE	DESCRIPTION
GameObject	

weight

Declaration

```
public float weight
```

Field Value

TYPE	DESCRIPTION
Single	