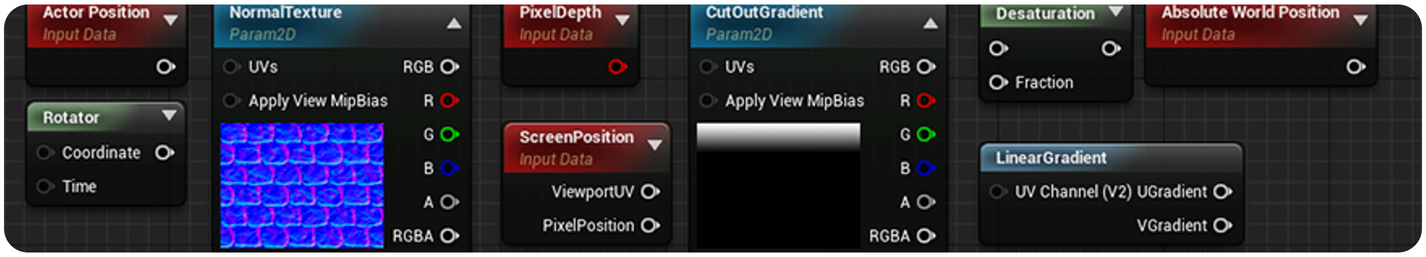
Material Parameter Expressions

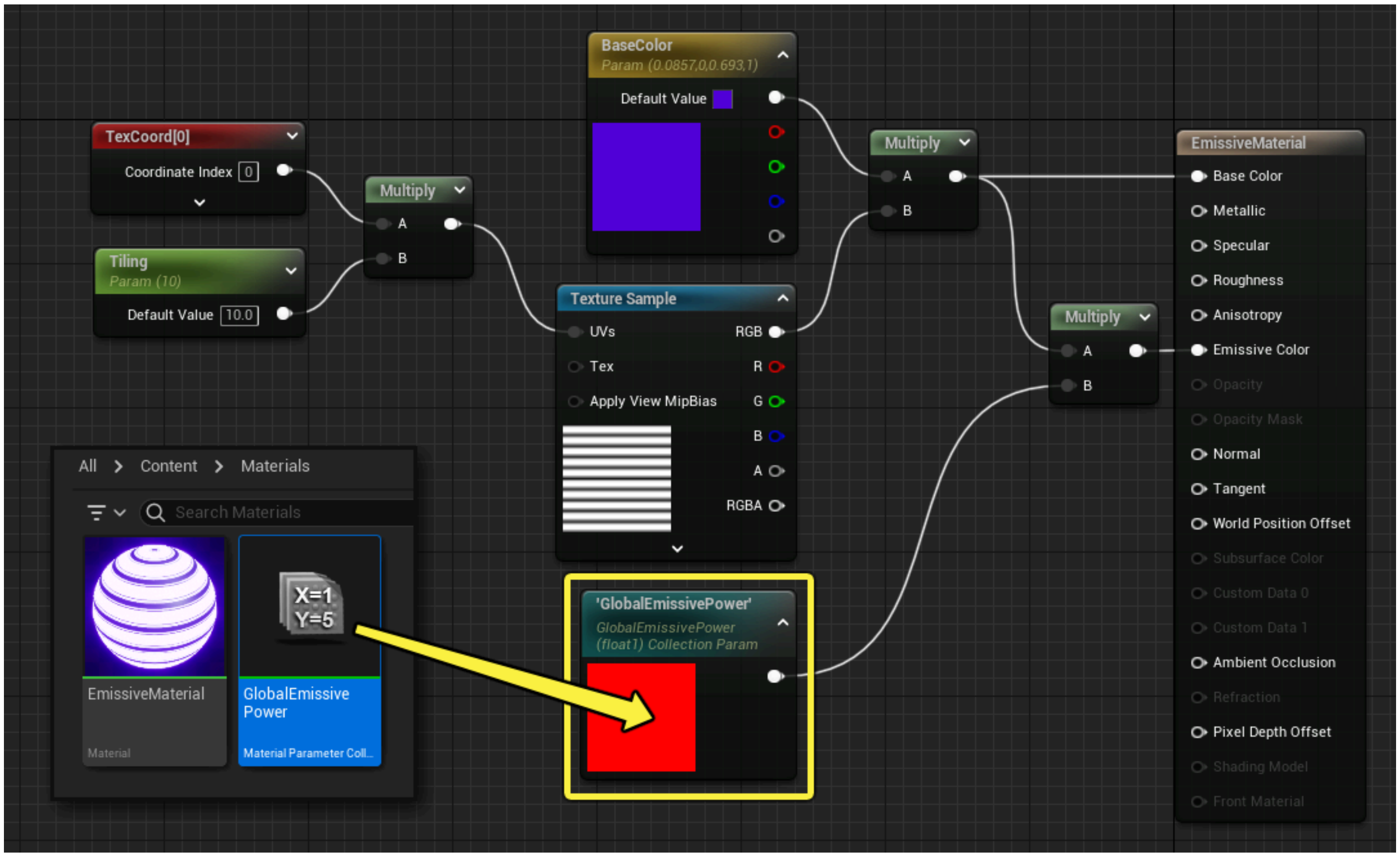
Expressions that expose properties to Material Instances to be overridden in child instances or modified at runtime.



Collection Parameters

A **Collection Parameter Expression** is used to reference a **Material Parameter Collection** asset. These are groups of parameters that you can easily reuse in many different assets such as Materials, Blueprints, and more.

They enable you to modify a global value once in the Parameter Collection, and have it propagate to mulitple Materials that reference the collection. For more information, read the Material Parameter Collections documentation.



Click to enlarge image.

A Material can reference, at most, two different MaterialParameterCollections. One is typically used for game-wide values while the other can be used for level specific parameters. A collection can have up to 1024 scalar parameters and 1024 vector parameters.

DynamicParameter

The **DynamicParameter** expression provides a conduit for particle emitters to pass up to four values to the Material to be used in any manner. These values are set in **Cascade** by means of a **ParameterDynamic** module placed on an emitter.

Property	Description
Param Names	An array of the names of the parameters. The values here will determine the text displayed on the outputs of the expression in the Material Editor and will be the names used to reference the parameters in the ParameterDynamic module in Cascade.

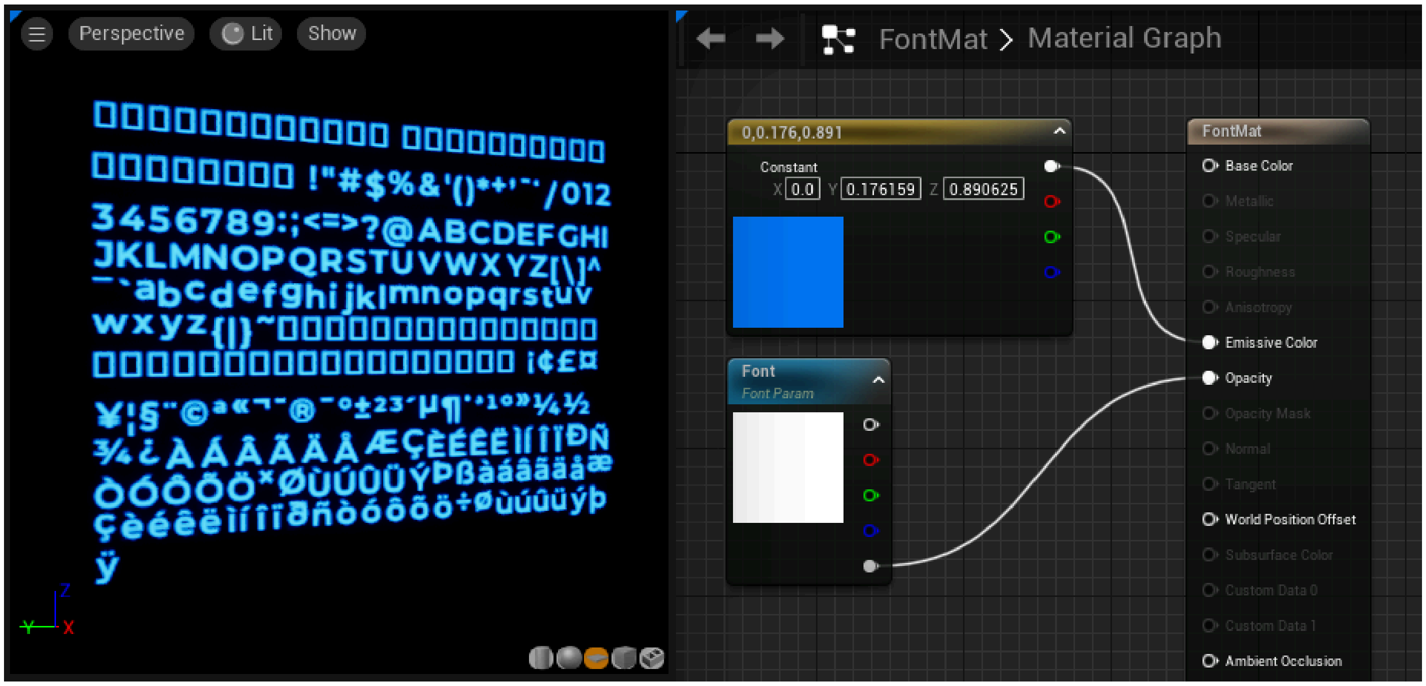
Property	Description
Default Value	Specifies the initial values that the parameter outputs (Vector4).
Outputs	
Param1	Outputs the value of the first parameter in the Param names property. The name of this output can change based on the values in the Param Names property.
Param2	Outputs the value of the second parameter in the Param names property. The name of this output can change based on the values in the Param Names property.
Param3	Outputs the value of the third parameter in the Param names property. The name of this output can change based on the values in the Param Names property.
Param4	Outputs the value of the fourth parameter in the Param names property. The name of this output can change based on the values in the Param Names property.

FontSampleParameter

The **FontSampleParameter** expression provides a way to expose a font-based parameter in a Material Instance Constant, making it easy to use different fonts in different instances. The alpha channel of the font will contain the font outline value. Only valid font pages can be specified.

Item	Description
Properties	
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that

Item	Description
	have the same Group property name will be listed underneath that category in the Instance Editor.
Font	Holds the default font asset (from the **Content Browser**) to be held within the expression.
Font Texture Page	The current font texture page to be used as a part of the texture.



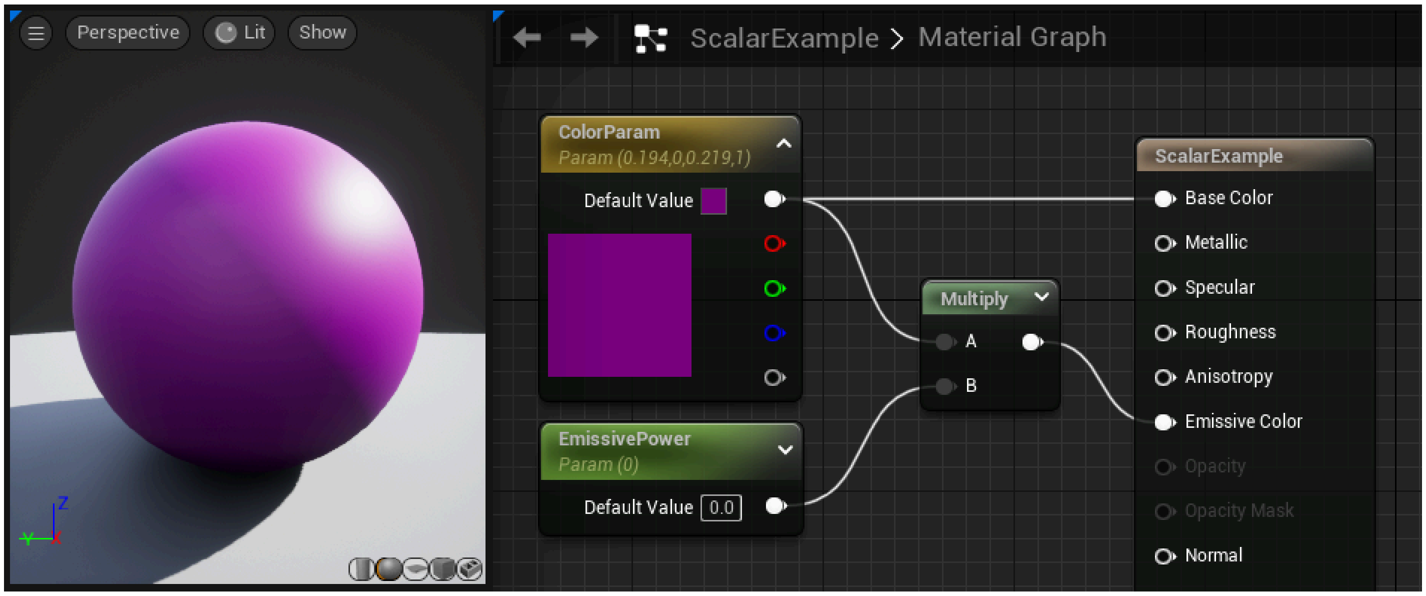
ScalarParameter

The **ScalarParameter** expression outputs a single float value (constant) that you can access and change in a Material Instance or on the fly by Blueprint or code.

Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that

Property	Description
	have the same Group property name will be listed underneath that category in the Instance Editor.
Default Value	Specifies the initial value that the constant takes on.

You can update the value in a Scalar Parameter, and immediately see the results without recompiling the Material.



Scalar parameter updated from 0 to 10.

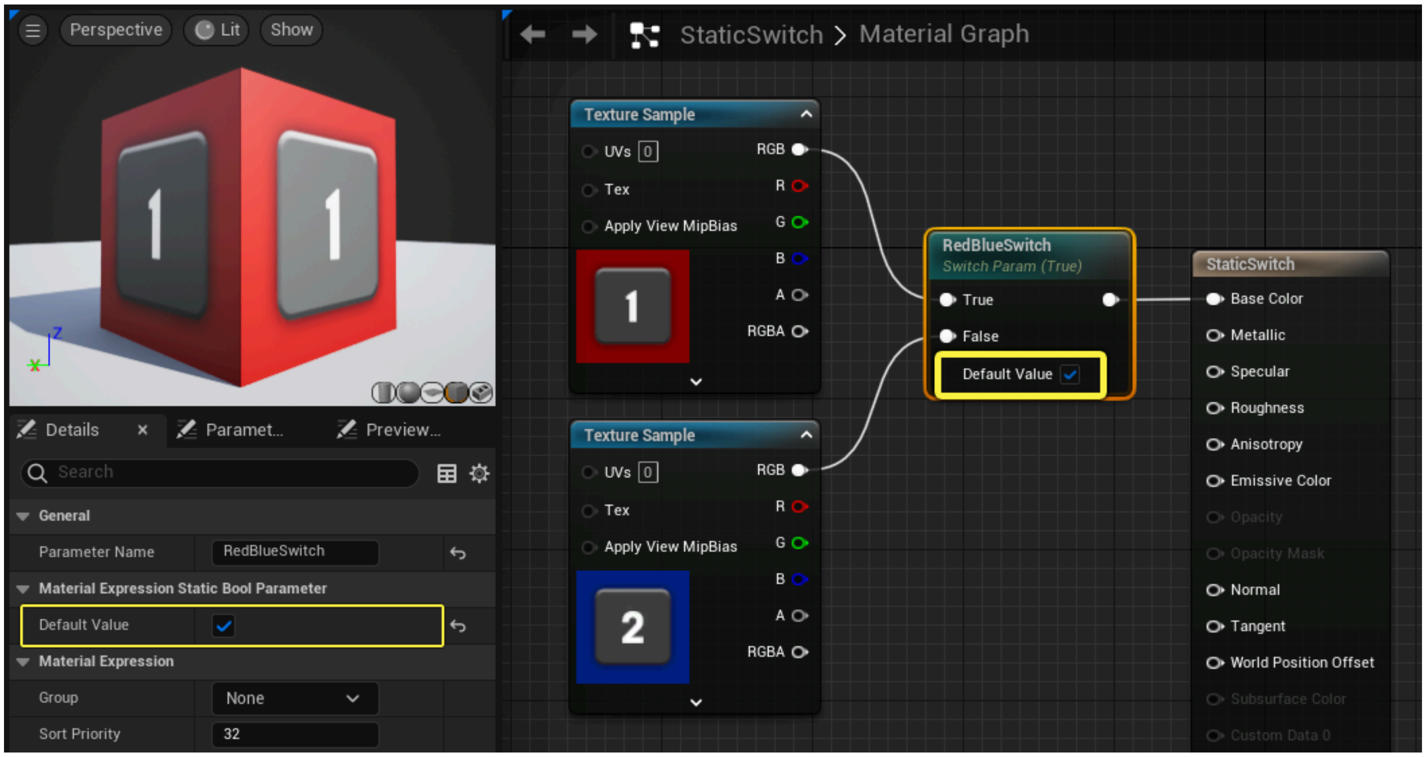
StaticSwitchParameter

The **StaticSwitchParameter** expression takes in two inputs, and outputs the first if the value of the parameter is *true*, and the second otherwise.

This parameter is named static because it cannot change at runtime; it can only be set in the Material Instance Editor. Static Switches are applied at compile time, not at runtime. This means that whatever branch of the Material was dropped is never executed, so static switches are effectively free at runtime. On the other hand, a new version of the Material must be compiled for every **used** combination of static parameters in a Material, which can lead to a massive increase in shader permutations if abused. Try to minimize the number of static parameters in the Material and the number of permutations of those static parameters that are actually used.

Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Default Value	If *true*, the first input is output. Otherwise, the second input is output.
Extended Caption Display	If *true*, the title bar of the expressions displays the value of the expression.
Inputs	
A	Takes in a value of any number of channels.
B	Takes in a value of any number of channels.

Example Usage: You can use Static Switches to remove an entire branch of a Material with no runtime cost. Material Instances can have different values, making it possible to have a templated shader setup with no performance loss.



General

Parameter Name	RedBlueSwitch

Material Expression Static Bool Parameter

Default Value	✓

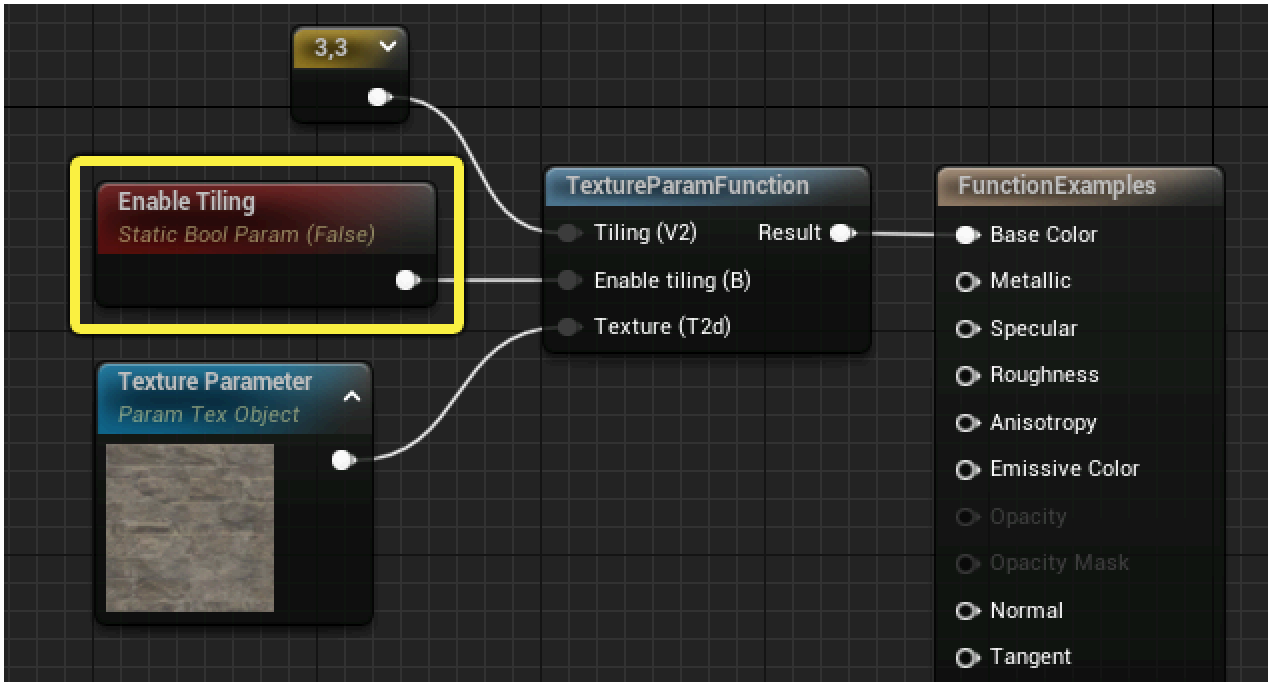
Material Expression

Group	None
Sort Priority	32

Static Switch updated from True to False.

StaticBoolParameter

The **StaticBoolParameter** works like a StaticSwitchParameter, except that it only creates a bool parameter and does not implement the switch. You can use a StaticBoolParameter to pass a default value into a boolean input on a Material Function.



Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Default Value	The default bool value of the parameter, *True* (checked) or *False*.

Static Switches are applied at compile time (not at runtime), meaning you can override the parameter in the Material Instance Editor, but it cannot change during gameplay. Whatever branch of the Material was dropped is never executed, which means static switches are effectively free at runtime. On the other hand, a new version of the Material must be compiled for every **used** combination of static parameters in a Material, which can lead to a massive increase in shader permutations if abused. Try to minimize the number of static parameters in the Material and the number of permutations of those static parameters that are actually used.

This node is used with MaterialFunctions.

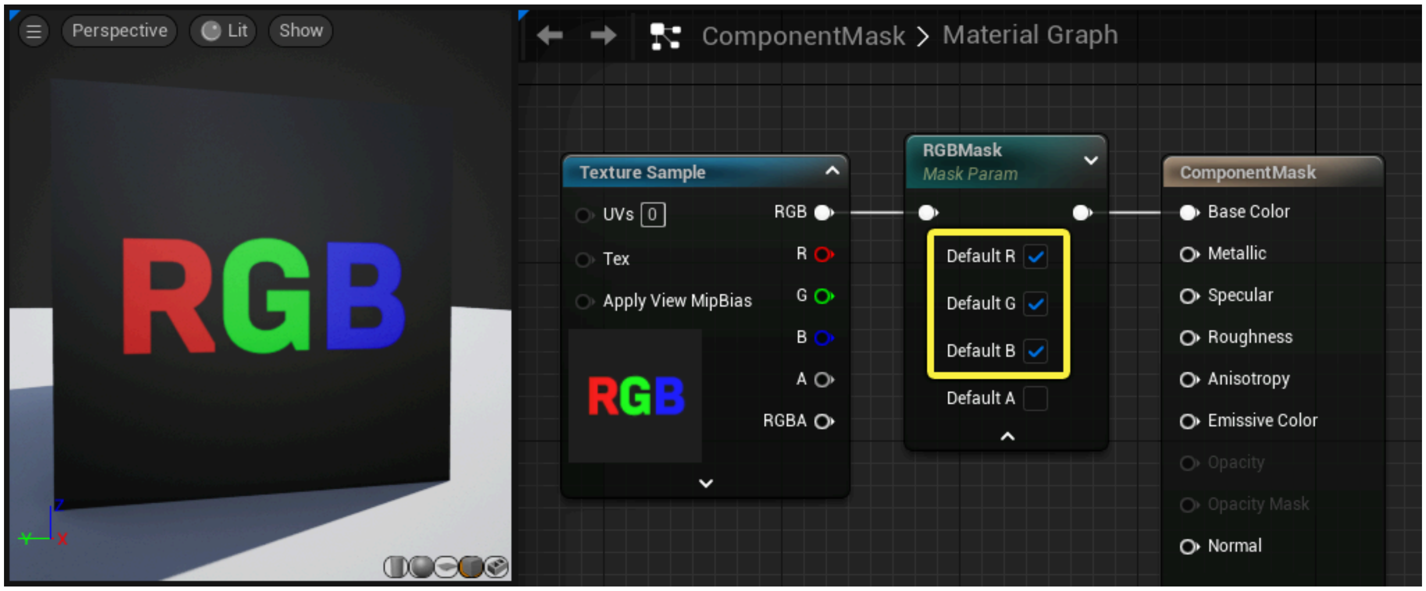
StaticComponentMaskParameter

The **StaticComponentMaskParameter** expression behaves like an ordinary Component Mask, except that you can override the mask values in a Material Instance.

This parameter is named static because it cannot change at runtime; it can only be set in the Material Instance Editor. Static Switches are applied at compile time, not at runtime. This means that whatever branch of the Material was dropped will never be executed, so static switches are effectively free at runtime. On the other hand, a new version of the Material must be compiled out for every **used** combination of static parameters in a Material, which can lead to a massive increase in shader permutations if abused. Try to minimize the number of static parameters in the Material and the number of permutations of those static parameters that are actually used.

Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Default R	If checked, the red, or first, channel of the input value is passed through to the output.
Default G	If checked, the green, or second, channel of the input value is passed through to the output.
Default B	If checked, the blue, or third, channel of the input value is passed through to the output.
Default A	If checked, the alpha, or fourth, channel of the input value is passed through to the output.

Example Usage: You can use Static Component Masks to let instances dictate which channel of a mask texture to use. If the mask is static (does not need to change at runtime) then this approach should always be used instead of multiplying a texture lookup by a Vector Parameter to mask out channels, since the latter wastes texture bandwidth and shader instructions.



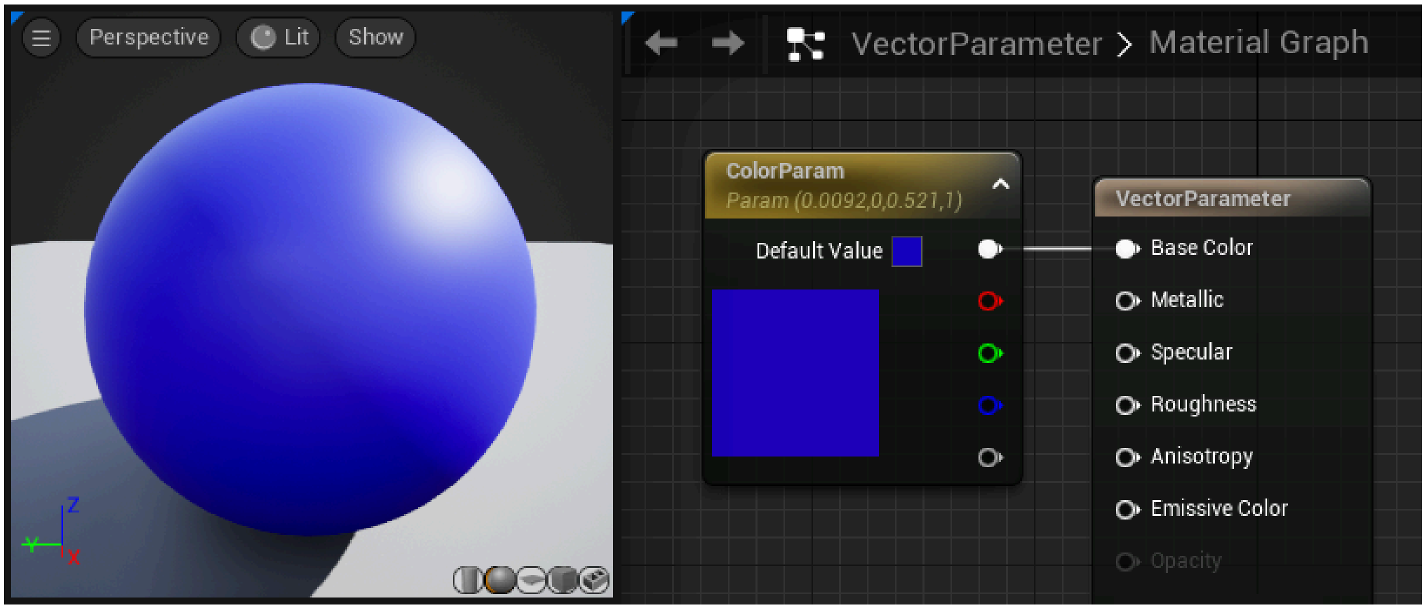
Checked channels pass through to the output, while unchecked channels are discarded.

VectorParameter

The **VectorParameter** expression is identical to the Constant4Vector, except that it is a parameter you can modify in instances of the Material and through code. One nicety of the VectorParameter is that you can set its value using the Color Picker.

Item	Description
Properties	
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.

Item	Description
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Default Value • **R** • **G** • **B** • **A**	The vector to output by default unless overridden by a Material Instance Constant. • Specifies the float value of the red, or first, channel of the vector the expression outputs. • Specifies the float value of the green, or second, channel of the vector the expression outputs. • Specifies the float value of the blue, or third, channel of the vector the expression outputs. • Specifies the float value of the alpha, or fourth, channel of the vector the expression outputs.

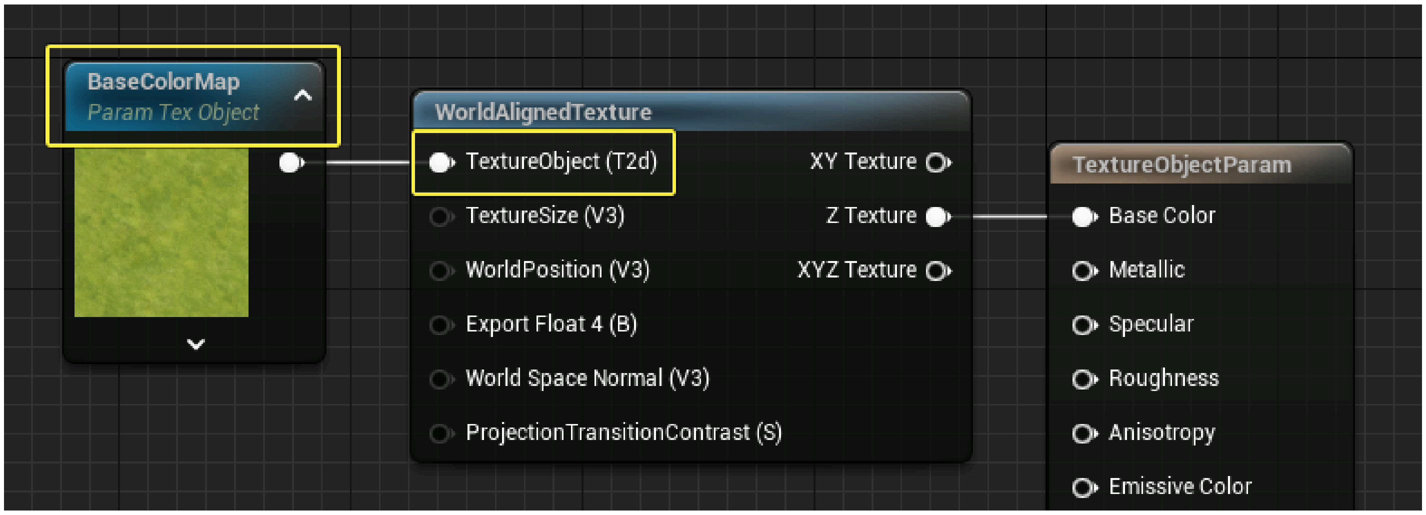


VertexColor is mutually exclusive with the Transform node due to limited interpolators. If you use both a Transform node and VertexColor, then VertexColor will come out all white.

Programmers: For sprite particles, colors are communicated to the shader per vertex, whereas colors for mesh particles are set as shader constants.

TextureObjectParameter

The **TextureObjectParameter** expression defines a texture parameter and outputs the texture object. This expression is frequently used to pass texture parameters into a Material Function with texture inputs. Texture inputs on a Material Function node are not compatible with Float3 data from a TextureSample 2D node, so the Texture Object (T2d) is required. This node does not actually sample the texture, so it must be used in conjunction with a **TextureSample** node.



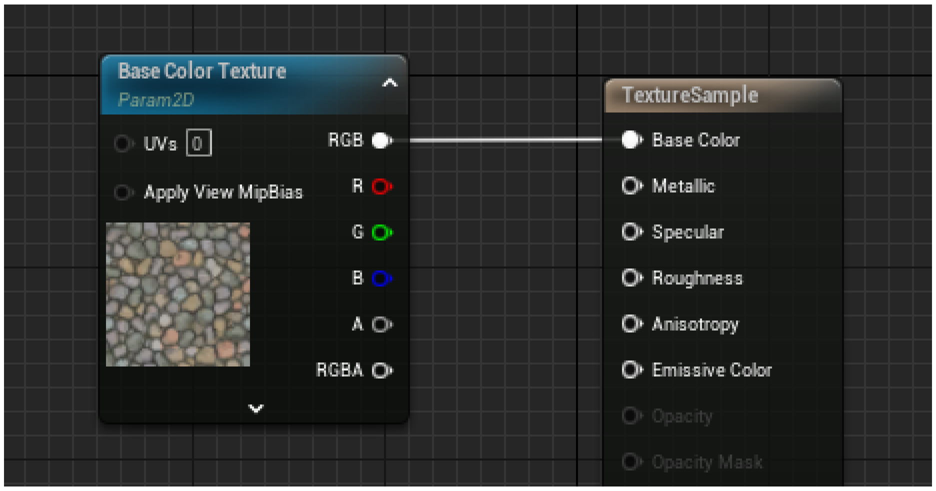
Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Texture	Specifies the texture sampled by the expression.
Sampler Type	The type of data that will be sampled and output from the node.

Property	Description
MipValueMode	Selects how to customize the sample's mip-level or derivatives from the default hardware computed. Affects the look and performance.

This node is used with MaterialFunctions.

TextureSampleParameter2D

The **TextureSampleParameter2D** expression is identical to the TextureSample except that it is a parameter that you can modify in Material Instances and through Blueprint or code.



Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Texture	Specifies the texture sampled by the expression.
Sampler Type	The type of data that will be sampled and output from the node.
MipValueMode	Selects how to customize the sample's mip-level or derivatives from the default hardware computed. Affects the look and performance.

Property	Description
Inputs	
UVs	Takes in UV texture coordinates to use for the texture. If no values are input to the UVs, the texture coordinates of the mesh the material is applied to are used.
Outputs	
RGB	Outputs the three-channel RGB vector value of the color.
R	Outputs the red channel of the color.
G	Outputs the green channel of the color.
B	Outputs the blue channel of the color.
A	Outputs the alpha channel of the color. If a texture does not contain an alpha channel, connecting the 'alpha' channel to something, while not technically illegal, will always result in zero (black).

TextureSampleParameterSubUV

The **TextureSampleParameterSubUV** expression is identical to the ParticleSubUV except that it is a parameter that can be modified in instances of the material and through code.

Item	Description
Properties	
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that

Item	Description
	have the same Group property name will be listed underneath that category in the Instance Editor.
Blend	Blends together each frame of the SubUV sprite layout, rather than instantly "popping" from one frame to the next.
Texture	Specifies the texture sampled by the expression.
Sampler Type	The type of data that will be sampled and output from the node.
MipValueMode	Selects how to customize the sample's mip-level or derivatives from the default hardware computed. Affects the look and performance.
Inputs	
UVs	The UV input is ignored and does nothing.
Outputs	
RGB	Outputs the three-channel RGB vector value of the color.
R	Outputs the red channel of the color.
G	Outputs the green channel of the color.
B	Outputs the blue channel of the color.
A	Outputs the alpha channel of the color. If a texture does not contain an alpha channel, connecting the 'alpha' channel to something, while not technically illegal, will always result in zero (black).

TextureSampleParameterCube

The **TextureSampleParameterCube** expression is identical to the TextureSample except that it only accepts cubemaps and it is a parameter that can be modified in instances of the material and through code.

Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Texture	Specifies the texture sampled by the expression.
Sampler Type	The type of data that will be sampled and output from the node.
MipValueMode	Selects how to customize the sample's mip-level or derivatives from the default hardware computed. Affects the look and performance.
Inputs	
UVs	Takes in UV texture coordinates to use for the texture. If no values are input to the UVs, the texture coordinates of the mesh the Material is applied to are used. This input accepts a two-channel vector value.
Outputs	
RGB	Outputs the three-channel RGB vector value of the color.
R	Outputs the red channel of the color.
G	Outputs the green channel of the color.

Property	Description
B	Outputs the blue channel of the color.
A	Outputs the alpha channel of the color. If a texture does not contain an alpha channel, connecting the 'alpha' channel to something, while not technically invalid, will always result in zero (black).

TextureSampleParameterMovie

The **TextureSampleParameterMovie** expression is identical to the TextureSample except that it only accepts movie textures (Bink movies) and it is a parameter that you can modify in instances of the Material and through code.

Property	Description
Parameter Name	Specifies the name used to identify the parameter in instances of the Material and through code.
Group	Provides a way to organize parameter names into groups, or categories, within a MaterialInstanceConstant. All parameters within a Material that have the same Group property name will be listed underneath that category in the Instance Editor.
Texture	Specifies the texture sampled by the expression.
Sampler Type	The type of data that will be sampled and output from the node.
MipValueMode	Selects how to customize the sample's mip-level or derivatives from the default hardware computed. Affects the look and performance.
Inputs	
UVs	Takes in UV texture coordinates to use for the texture. If no values are input to the UVs, the texture coordinates of the mesh the Material is applied to are used.

Property	Description
Outputs	
RGB	Outputs the three-channel RGB vector value of the color.
R	Outputs the red channel of the color.
G	Outputs the green channel of the color.
B	Outputs the blue channel of the color.
A	Outputs the alpha channel of the color. If a texture does not contain an alpha channel, connecting the 'alpha' channel to something, while not technically illegal, will always result in zero (black).