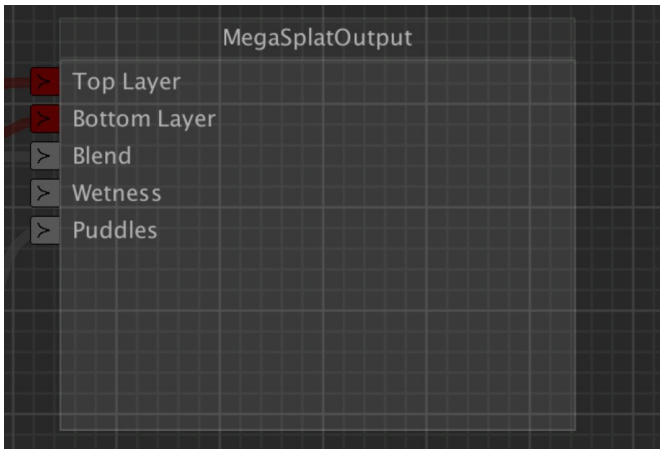
MegaSplat Texture Graph

The MegaSplat texture graph is a procedural texturing tool for texturing meshes and terrains with a shader graph style interface instead of painting. This can be an extremely useful way to create an initial paint of a large world and persist that paintjob through rapid geometry changes.

If you are familiar with shader graphs, then you will feel at home in the Texture Graph. In a shader graph, the base node contains inputs to the lighting equation; things like diffuse color, normal, and various specular properties. The root node in the Texture Graph contains inputs into the MegaSplat technique. Understanding how MegaSplat blends textures is at the core of using the graph. With a two layer shader, each control point (vertex or pixel when using a terrain) stores two texture choices and a blend weight between them. Inputs for wetness and puddles are also available.

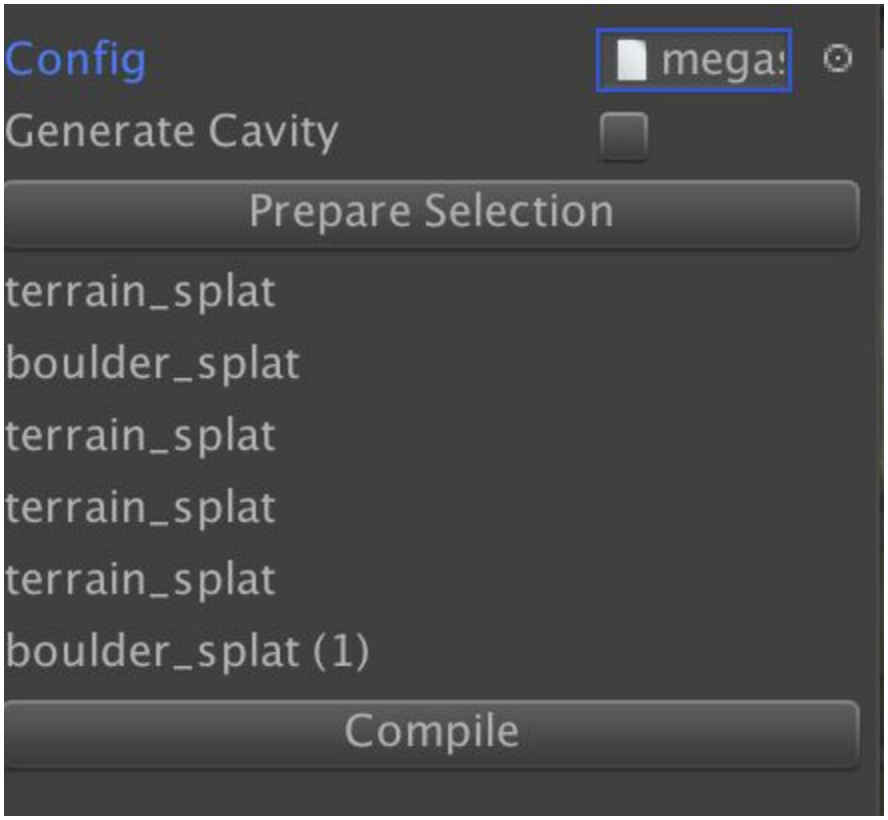


Note that texture choices are represented by red connections, and float values by grey. These are the only two types needed by the Texture Graph.

Quick Tutorial

Open the graph from the Windows->MegaSplat->Texture Graph menu option and select New Canvas->Default with the button on the right. Load the example mesh terrain and select the meshes game object.

In the Texture Graph, press “Prepare Selection”. This will go through the current selection and prepare any meshes with VertexInstanceStreams on them or any MegaSplat terrains for processing.

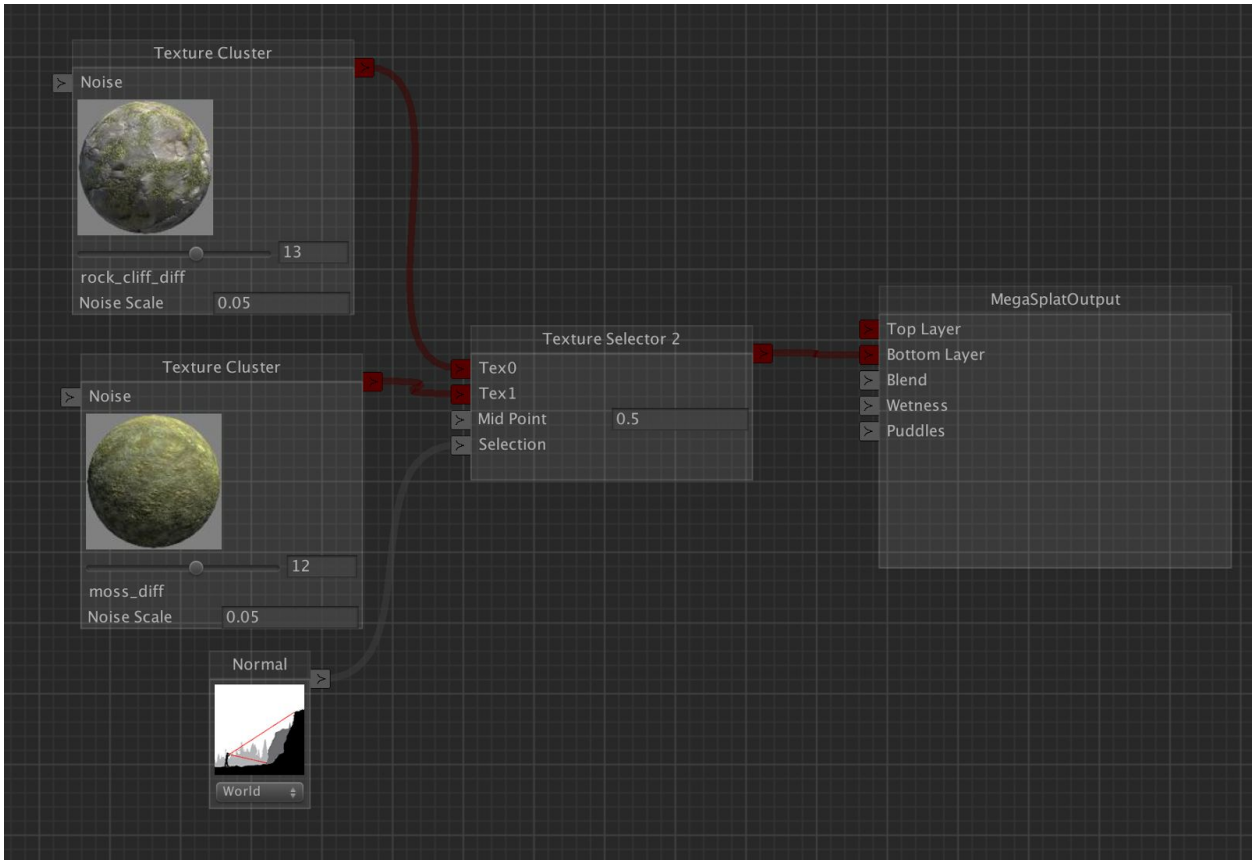


Once prepared, the UI will list all objects which are going to be processed. Finally, select the “config”, in this case, the diffuse Texture Array Config that ships with MegaSplat. We are now ready to start texturing using the graph.

Right click in the graph window and select “Add MegaSplat->TextureCluster” from the drop down list. Use the slider to select the “moss_diff” cluster, or use the numeric field and set it to 12. Create a second Texture Cluster node and select the 13th cluster, “rock_cliff_diff”.

What we are going to do is texture flat areas of our terrain with grass, and the sides of hills with the rock_cliff cluster. To do this, we need to create a “selector2” node (“Add MegaSplat->Sector2”. A selector2 node will compare a value and select between two textures based on that value. Click on the red out port next to the rock_cliff cluster and drag a line to the tex0 input on the selector2 node. Then connect the moss’s output node to the tex1 input on the selector2 node.

Next, add a “Add Data->Normal Angle” node and connect it’s output to the “Selection” input on the selection2 node. Finally, connect the output of the selector2 node to the bottom input on the MegaSplat Output node.



Here is our first completed graph. What we are doing is deciding between two texture clusters based on the angle of the normal.

Press “Compile” to compile the graph. Once the graph is compiled, any value changes in the graph happen in realtime. Note that any changed to the graph flow will require a recompile to take effect.

Once you hit compile, your terrain should become entirely filled with moss. No rock_cliff was used, because the midpoint on our selector2 is too low. Change the value on the selector to 0.7 and rocks will begin to appear on the slopes. You can continue to tweak the value until you are happy with the result.

Bring in a second layer

The mesh example scene uses a two layer MegaSplat shader. Let’s provide some variation to this scene by bringing in a second layer.

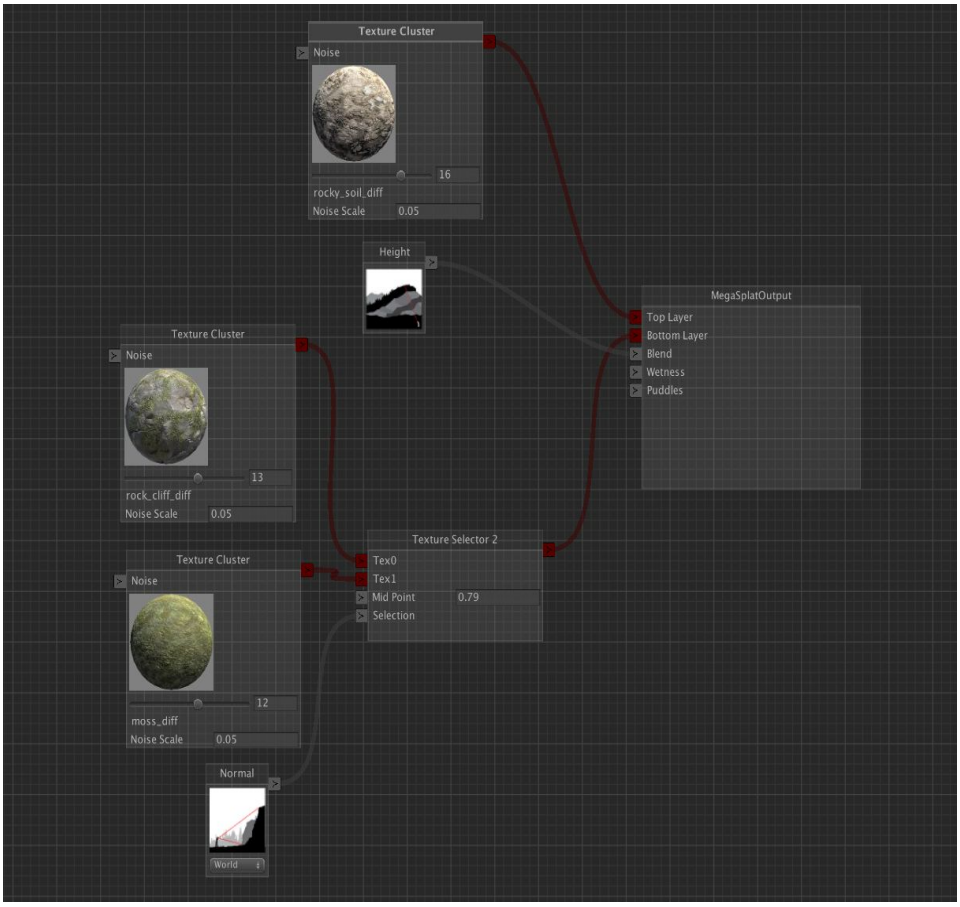
Create another texture cluster and select number 16, “rocky_soil_diff”. Plug this directly into the top layer of the output node.

Now create a ‘height’ node (“Add Data->Height”), and plug it directly into the blend. Press compile to see the result.

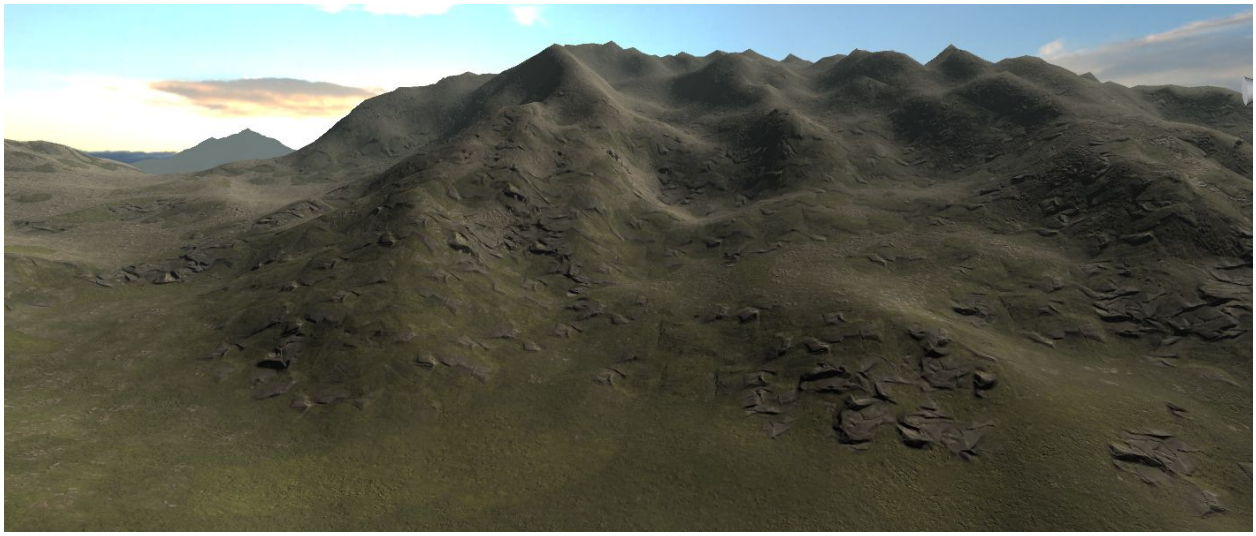
The height node outputs a 0-1 value representing the height of the object. In this case, we have told the graph to blend towards our original moss/rock texturing at the bottom, and then

cross fade into using the new texture at the top. Note that the height map blending makes this look good without any real adjustments.

You're graph should look like this:

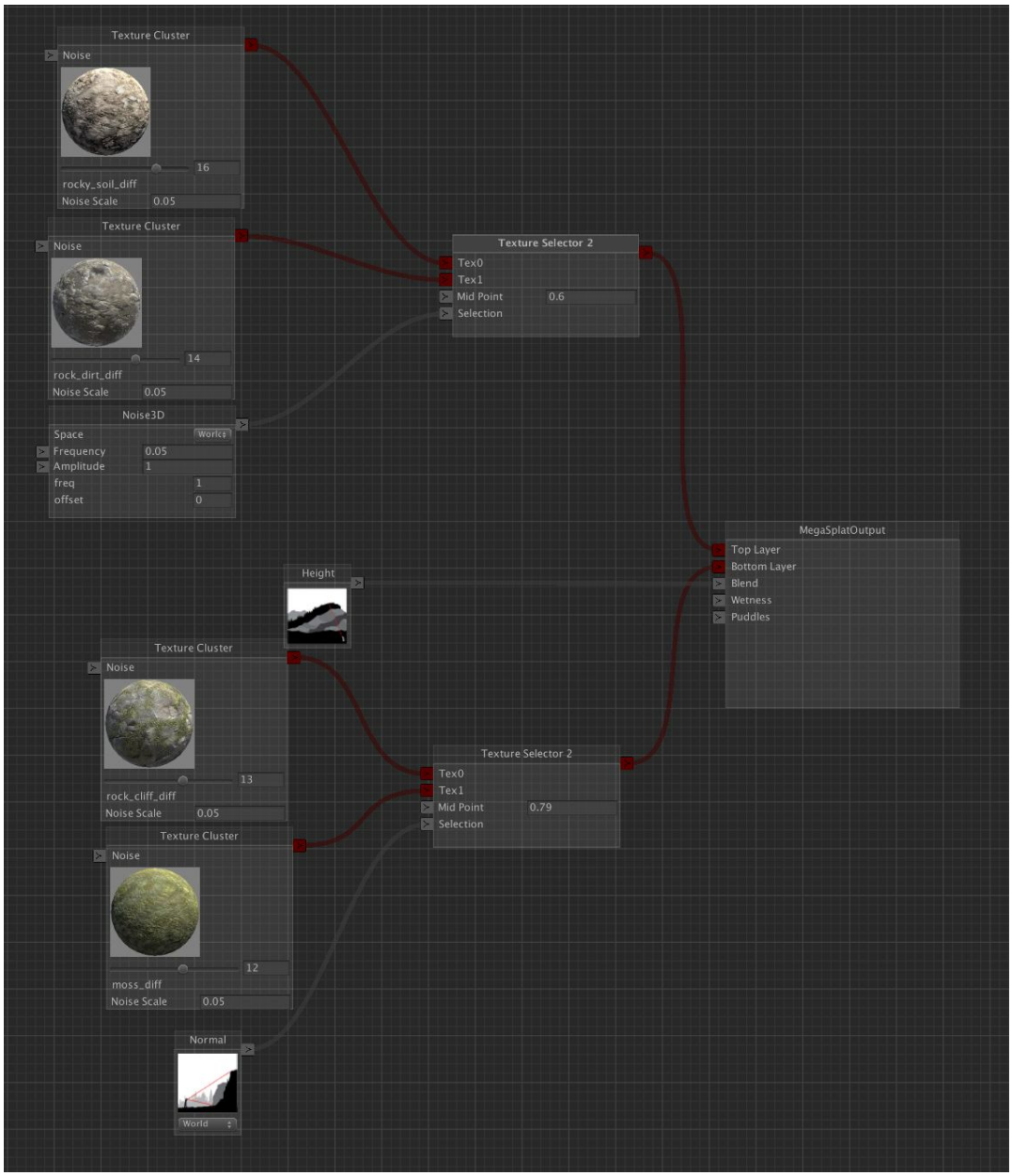


And your terrain should look like this:



Let's take it a step further. The sand at the top of the mountain is a bit to regular for my tastes. Let's sprinkle some small stones in there to break it up a bit.

Create another texture cluster and select cluster 14, rock_dirt. Create another selector2, and wire clusters 16 and 14 into it. Now create a noise node “Add MegaSplat->Noise”, and plug the output of the noise node into the selector2’s selection input. Plug the output of the selector2 into the top layer input. Compile. Now adjust the selector2’s midpoint value to control the ratio of each texture. This is, in essence, a texture cluster of 2 texture clusters! You can adjust the noise node’s parameters to control how often it switches between these two textures. Here’s a shot of the current graph:



If you explore you terrain, you’ll notice small bits of rock coming through the entire terrain, creating a rich texturing with only a few nodes. We’ve only touched on the capabilities,

but have a complex and pleasing result that would be impossible to achieve with other systems in just minutes of work.

Take some time to explore the example graphs in the example folder, and read through the node documentation to understand what each node does. Note that if you are using a Unity Terrain, you will have to save the terrain textures via the save button in the Terrain Painter window.

Node Guide

Data Nodes

Data nodes provide input from external sources to the graph.

Cavity

The Cavity Node provides information about how concave or convex a given surface is. When a surface smooth, it has a value of 1. When a surface is concave or convex, the value gets lower.

Cavity needs to be turned on in the main window, and the meshes/terrains need to be prepared with cavity on. Computing the cavity can take some time, which is why it's optional. When computing cavity, you can determine how many samples should be taken per control point, and how wide the sampling area is.

Height

The height node outputs the height of the control point as a 0-1 value. The lowest point of the working object set will be at 0, and the highest at 1.

Noise3D

The noise node generates a 3 octave 3d noise value based on the local or world position of the control point. Noise frequency and amplitude are available. While rulesets are the best for making choices, coherent noise produces pleasing patterns, and can be used to select textures or break up rules which are to clear.

Normal Angle

Outputs the world or local space normal vs the Up vector as a 0-1 value. The up vector is specified as well, so if you'd like to compare the normal vs another vector you can.

TextureMask

Uses the UVs of the objects to sample a texture. Output's 4 float values (one for each texture channel). This can be used with masks generated via external programs, or painted.

Filter Nodes

Filter nodes allow you to modify and filter float values into new values. As an example, you can use a curve to remap any input value into an entirely different output value.

Pass Filter

This is essentially a low or high pass filter, if you familiar with Audio. You select the frequency and blend width, and the min/max values can be reversed to switch between a low pass or high pass filter. The basic idea of these types of filters is that they remap any value below a certain value to the min, and any value above it to the max value, with a small blend region in between.

An example of it's use would be to filter the height such that something only appears in the low areas.

Band Pass

A Band pass filter is like a high and low pass filter combined. It can isolate a specific area in the spectrum. As an example, you could use it to only apply textures between .1 and .3 of the terrain height.

MultiPass

The MultiPass filter is a 1-4 stage band pass filter with built in texture selection. The idea is that you can isolate 4 ranges, each with their own band pass filter, select textures and produce a blend weight for each all in one node.

For instance, you might want rock to appear at 65% weight on the top, cliff to appear at a 50% weight in the middle, and sand to appear with an 80% weight at the bottom. This is easily created and edited with the Multi-pass filter, which outputs a final texture choice and a blend weight (which can be used in the blend input of the master node).

Remap Curve

The Remap Curve node gives you an arbitrary curve in the 0-1 space, allowing you to remap any 0-1 input (height, noise, angle, cavity, etc) to any arbitrary 0-1 value. Note that all the other pass filters can be created with the Remap Curve, but in some cases it's easier to work with the other filter nodes rather than modifying curve data directly.

MegaSplat nodes

Texture Index

The texture index lets you select a single texture.

Texture Clutser

The texture cluster node lets you select a texture cluster and get the resulting texture from it. It is the equivalent to a noise function, some number of Texture Index nodes, and selectors to choose based on the noise output.

Selector2, Selector3, Selector4

The selector nodes give you an easy way to select between several texture choices. You can control the mid points for each selection, allowing you to weigh the choices.

Other Nodes

A small collection of basic math nodes is provided.

A comment and Float value node are also provided, should you want to comment your graphs or provide a constant float value to another node.