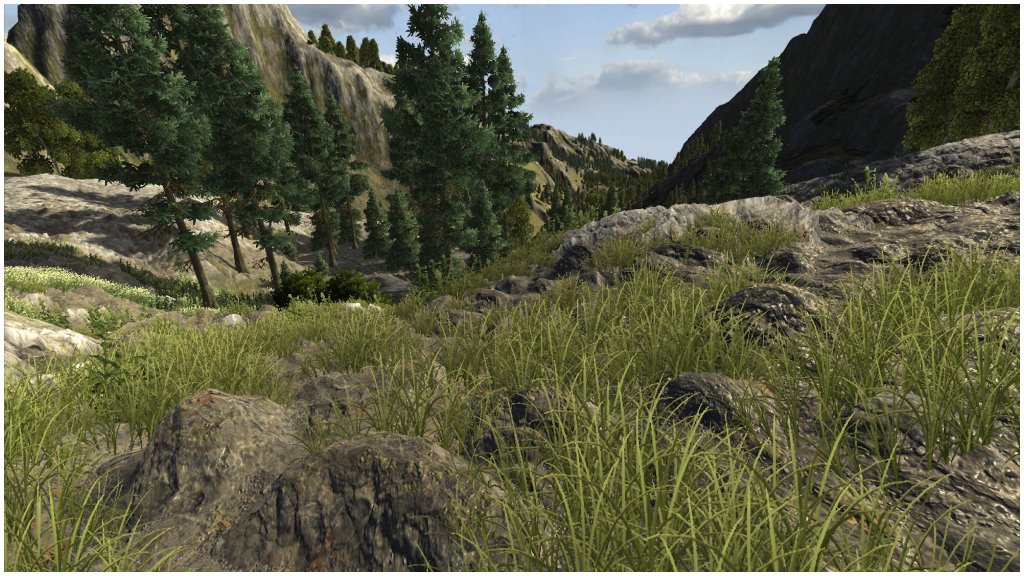
MicroSplat

Core Module Documentation



Overview

MicroSplat is a modular shading system, allowing you to add new features with new modules. At its core is the fastest splat mapping shader available, supporting up to 32 textures, and is automatically generated based on the settings you choose in order to provide the most optimal shader possible. This unique shader compiling framework allows for MicroSplat to have more options than a traditional shader would provide, and doesn't require you to select from certain combinations of features, only to find that the feature combination you need is not available.

The major focus points of MicroSplat are designed around:

- Performance
- Extremely High Quality rendering
- Ease of Use
- Modularity
- Massive Feature Set

Resources

Documentation for **every module** is included in **Packages/MicroSplat-Core/Documentation** - please read it.

Documentation for **every module** is included in **Packages/MicroSplat-Core/Documentation** - please read it.

Documentation for **every module** is included in **Packages/MicroSplat-Core/Documentation** - please read it.

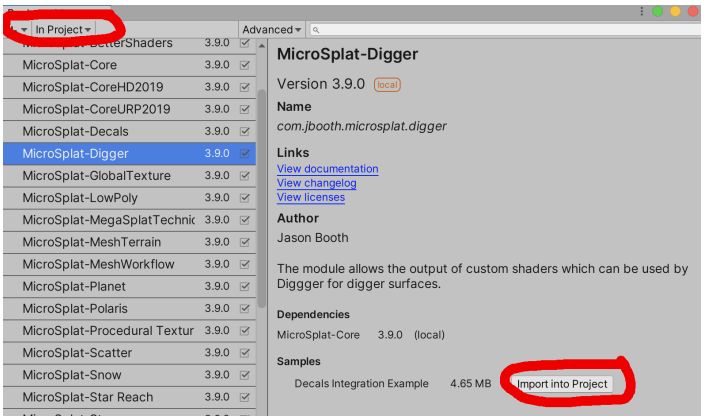
And just in case, sing it with me now, **Documentation** for **every module** is included in **Packages/MicroSplat-Core/Documentation** - please read it.

Videos for each module's features are included with the asset description of each model, and are all available on my YouTube channel:

YouTube Channel

Packages

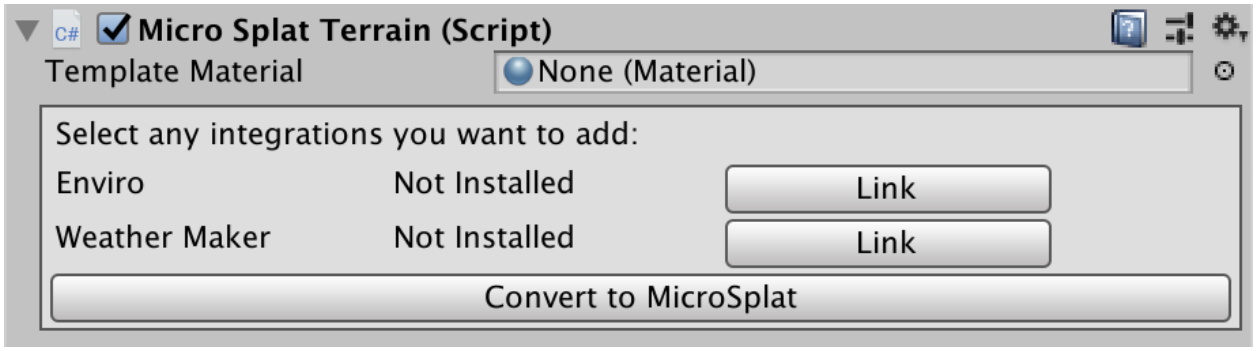
Note that MicroSplat installs itself as a local package. This means you can find it's data under the Packages folder, not the asset folder. All samples can be accessed from the package manager by selecting the "In My Project" tab and installing the sample for each package.



Getting Started

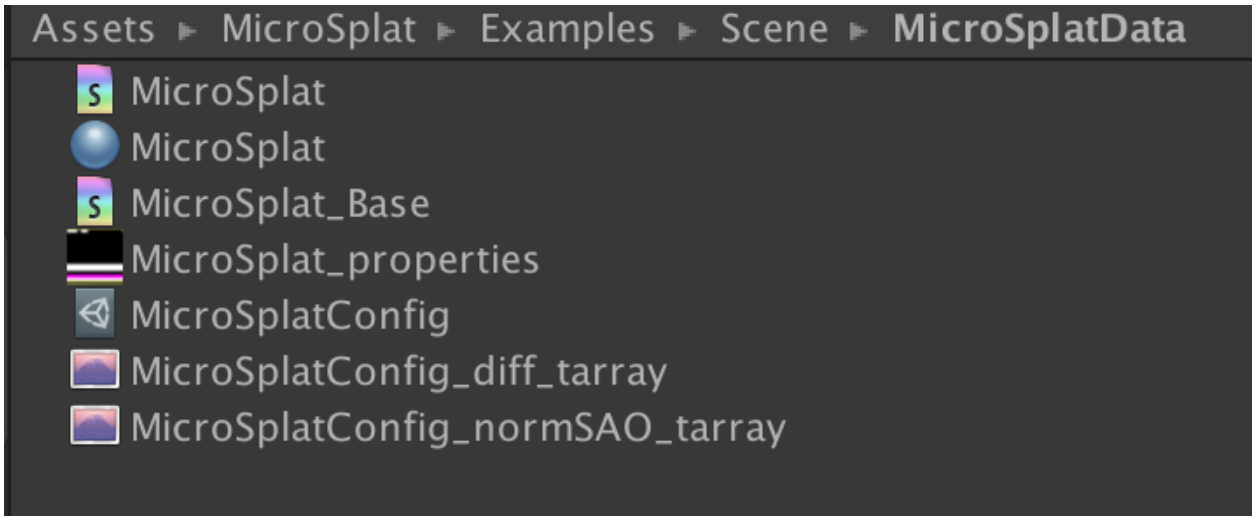
If you have an existing Unity terrain, getting started is only a few clicks.

- Select all the terrains you want to treat as a single, continuous terrain.
- Click “Add Component” in the inspector window, and select the “MicroSplatTerrain” component.



- You can add or enable integrations with 3rd party tools, or just leave everything as default.
- Press the “Convert to MicroSplat” button

In just a few seconds, this will read the existing textures from your terrain and pack them into custom texture arrays, and create a template material and shader. A full PBR workflow is generated from the texture maps you have available. For instance, if you only use diffuse maps, the MicroSplat will generate a Normal Map, Height map, Smoothness and Ambient Occlusion map for your textures automatically.



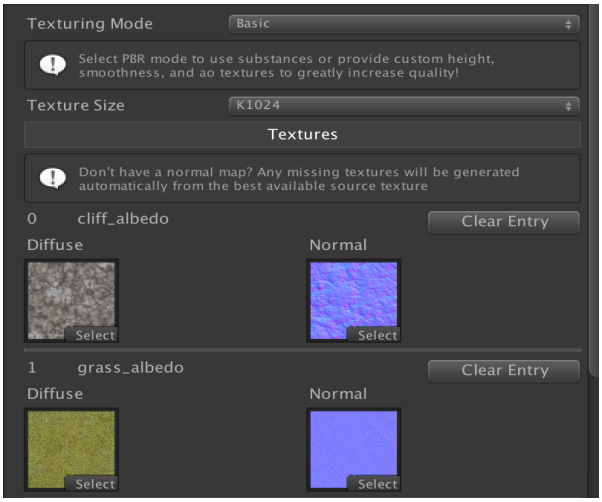
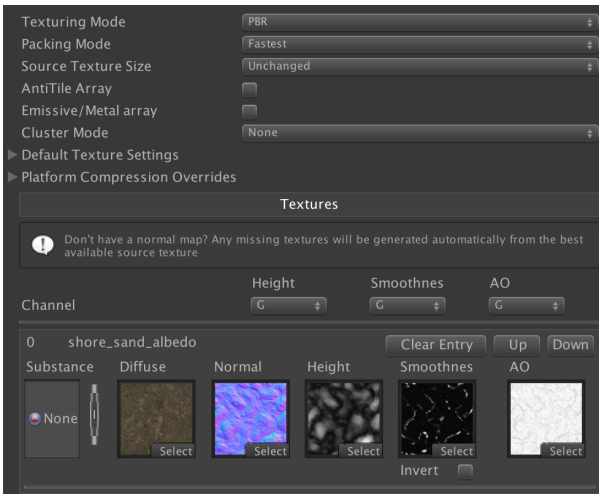
All data created will be organized into a MicroSplatData directory located next to your terrain data. In general, it's best to not rename this data, as some data has to be associated by name (materials cannot, for instance, have references to objects).

The Texture Packer

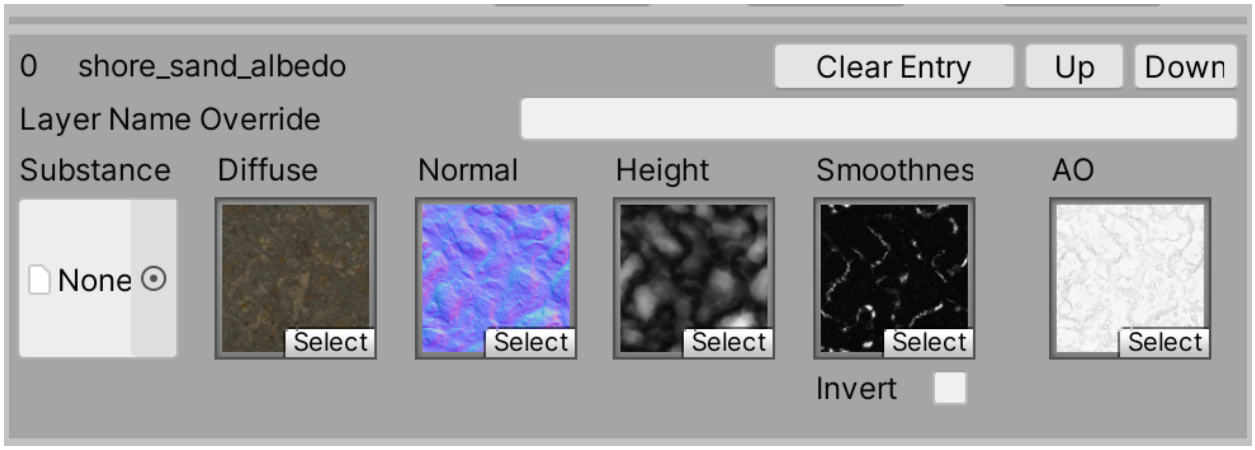
Once the conversion is done, the Texture Array Config will be opened. This is where you will manage textures for your terrain from now on. (You can come back to this screen by re-selecting the TextureArrayConfig object in the MicroSplatData directory.) There are two modes to this interface; a basic version, which only has slots for diffuse and normal maps, and a

PBR mode, which allows for substance importing, a full PBR workflow, and more controls over the resulting textures.

Note that no matter which mode you select, any missing textures will be automatically generated from the best available texture as the data is packed into the resulting texture arrays (they don't show up in the interface, or get written to disk). However, the best quality will always be obtained by providing high quality PBR textures, with all texture data defined. High quality normal maps and height maps have the biggest effect on shading quality.

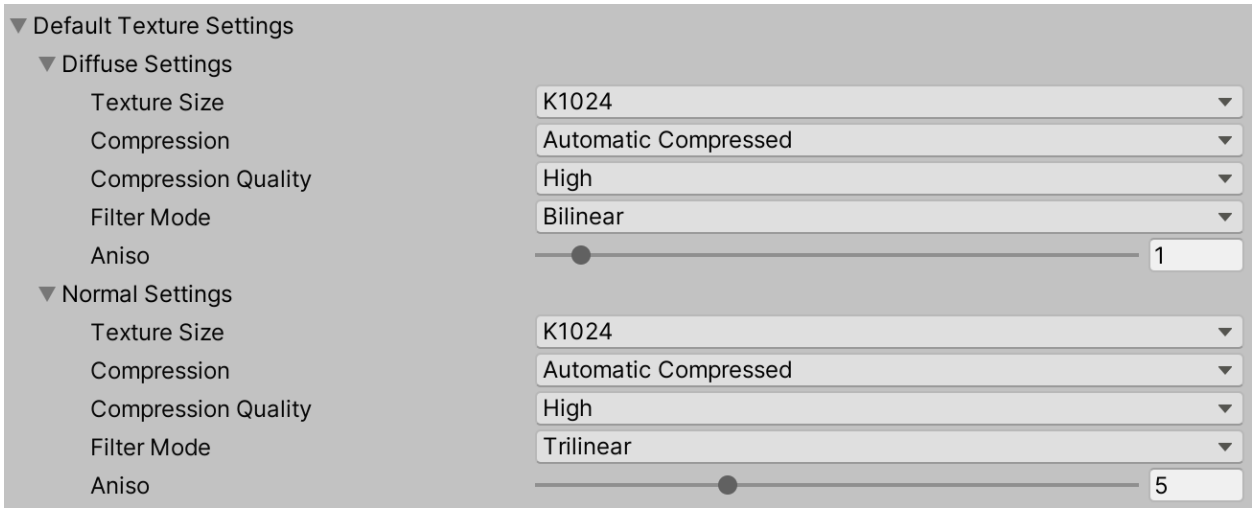
<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Basic Mode: A single resolution for all textures is exposed, and only diffuse and normal maps are available. All other channels are generated automatically.</td>
<td>PBR mode: More control over texture formats, substance importing, and the ability to provide all PBR texture channels. You can even choose which channel to pull the data from, or invert the smoothness map if your texture software provides roughness maps instead of smoothness maps.</td>
</tr>
</table>

If you make any edits, press the Update button at the bottom of the interface to regenerate the texture arrays.



Each entry in the Texture Array Config will have a number of textures and controls for data management. This will change based on what features are available. Note that by default, MicroSplat will create terrain layers for the texture data based on the texture name and index into the array. However, if you want to override these names with custom file names, you can enter the name you want to use in the Layer Name Override. Note that you are responsible for keeping these names unique - if you name every layer “grass”, or have multiple arrays in the same MicroSplatData directory with “grass” as the layer name, MicroSplat will only create one layer for them.

Texture Settings



Texture settings are available for each array type, allowing you to control the final output formats. A platform override array is also available, allowing you to specify these settings on a per platform basis.

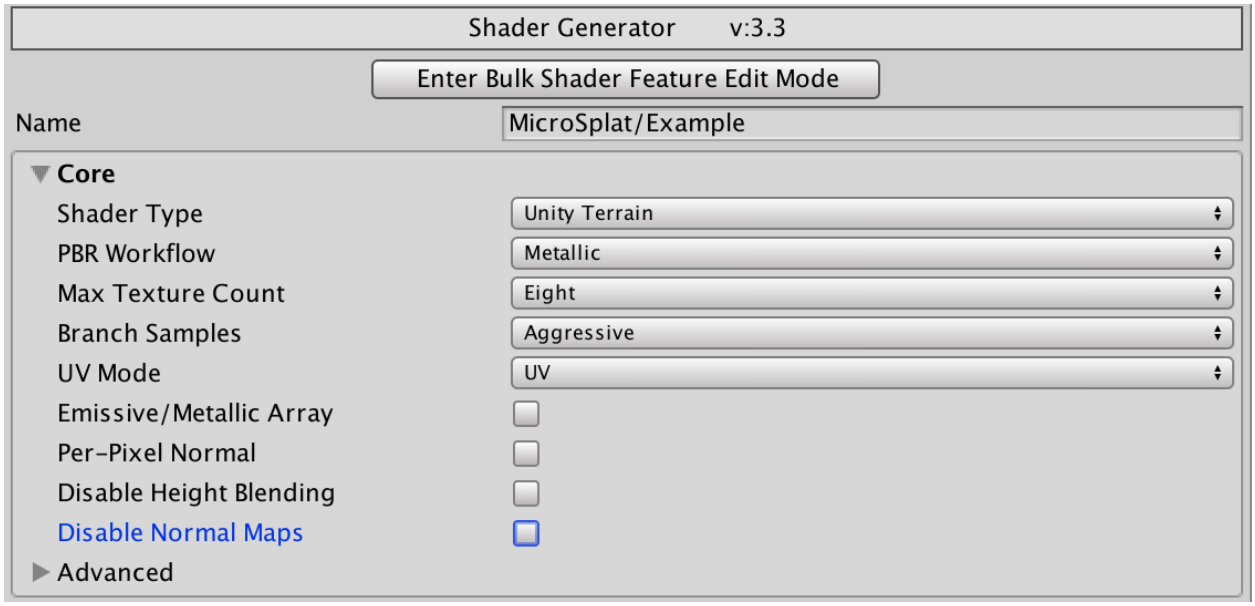
Unity's tools and API often make use of the source textures assigned to the terrain. MicroSplat leaves these textures alone by default, but a Source Texture Size property is available allowing you to reduce the resolution of these textures automatically. This can save significant memory in builds.

Note that compression quality maps to different formats on some platforms. For instance, on PC/Mac, compression quality of high maps to BC7 format, while medium and low map to DXT5. When using ATSC compression, the settings map to 4x4, 6x6, and 8x8 block sizes. When using PVR compression, low maps to 2bit per block, while the other settings use 4 bits.

The Shader Generator

To quickly access the shader settings, select the terrain and double click the ‘material template’ property on the MicroSplatTerrain component. This will open up the material which will be used by your terrains.

At the top of the shader’s UI is a list of currently available modules for MicroSplat. Clicking on one of these modules will bring you to the asset store where the module can be purchased. Once you download and install the module (just like any other Unity Asset Store asset), new features will become available in the Shader Generation section of the shader UI.

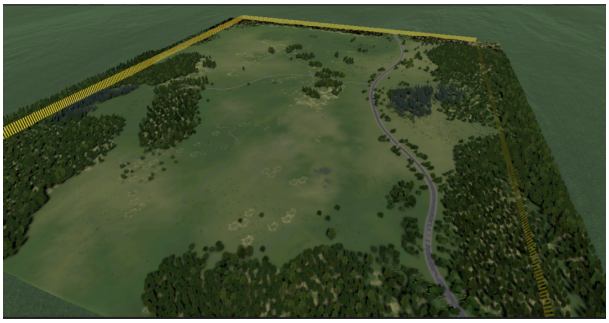
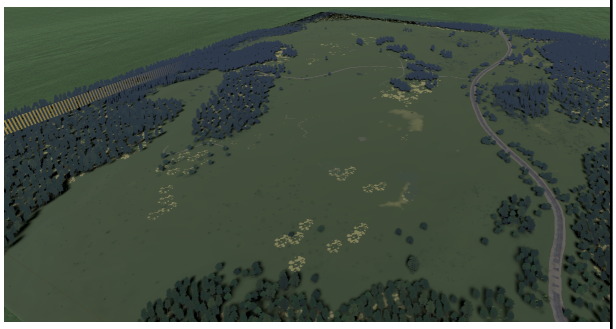
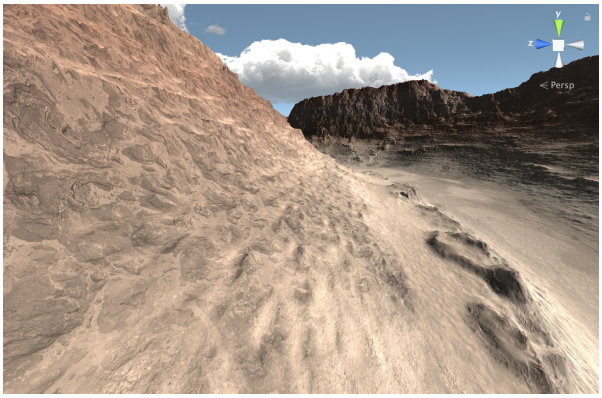
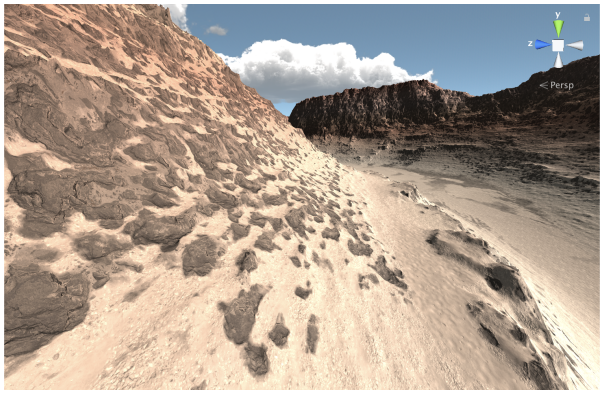


Changing any option in the Shader Generator section causes the shader to be recompiled with the new code included. As you add extension modules, the number of options available in this section will grow substantially, each listed under the module that provides the options. The options available for core include.

- **Shader Type**. This option only shows when you have modules that at new shader types, such as the Mesh Workflow Or Mesh Terrain module. You usually don't need to edit these by hand, as you usually use the setup systems for each of these types of shaders.
- **PBR Workflow**. This option allows you to choose between a metallic and specular workflow. Note that you will have to setup your texture arrays in this same workflow and assign the appropriate arrays to the material for the additional texture arrays.
- **Max Textures**. This controls how many textures the shader supports on the terrain. It defaults to 16, but you should use the lowest value you can which is above the number of textures you're using on your terrain.

- **Branch Sampling**. When this is enabled the shader will use Dynamic Flow Control to reduce the number of samples taken at each pixel. This can be a massive win with features like Texture Clustering, Stochastic Sampling, Triplanar Projection, or Distance Resampling on. Usually you can set this to aggressive for best performance and forget about it. Please see the included Optimization guide for more information on this setting, or this video: http://shorturl.at/quNPS
- **UV Mode**. By default, the texture maps will use the UVs from the terrain, but when you set this to WorldSpace, it will use a world space projection for the splat map UVs, which can be useful if you want multiple terrains to seem perfectly with oddly scaled UVs.
- **Emissive/Metal Array**. Allows you to use an extra texture array with a per-pixel emissive and metallic map packed together. You can enable the creation of this array in the TextureArrayConfig and assign the resulting array after enabling this option.
- **Per Pixel Normal.** In Unity 2018.3, unity added per-pixel normal support for terrain. This uses a normal map to store the normal for the terrain and sampling it per pixel instead of computing it on the GPU per vertex. The main advantage to this is better looking normals which are not affected by the changing vertex count. However, Unity only supports this option when drawing with instancing on, which currently is incompatible with tessellation in the standard pipeline, and per pixel normals were not available for older versions of Unity. Per Pixel normal is automatic in 2018.3 when instancing is enabled, but in previous versions of Unity or when instancing is not used, you can turn on this option to enable it. Once enabled, an option to bake the normal map is available in the MicroSplatTerrain component, and must be updated whenever the terrain topology changes.
- **Texture Blend Mode**
    - **Height Blended**. In this mode, the height map is used to blend between textures, and interpolation contrast controls how soft or hard the blend is
    - **Unity Linear**. Blends like Unity terrain, very blurry
    - **Hybrid Height Linear**. Blends between height blending and Normalized Linear Blending based on distance. This lets you get soft blends in the distance, but tight height mapped blends up close.

- **Normalized Linear**. Blends like Unity Terrain, but makes sure the weights total one as to prevent over brightening.

Linear Blend	Height Map Blend
	
	

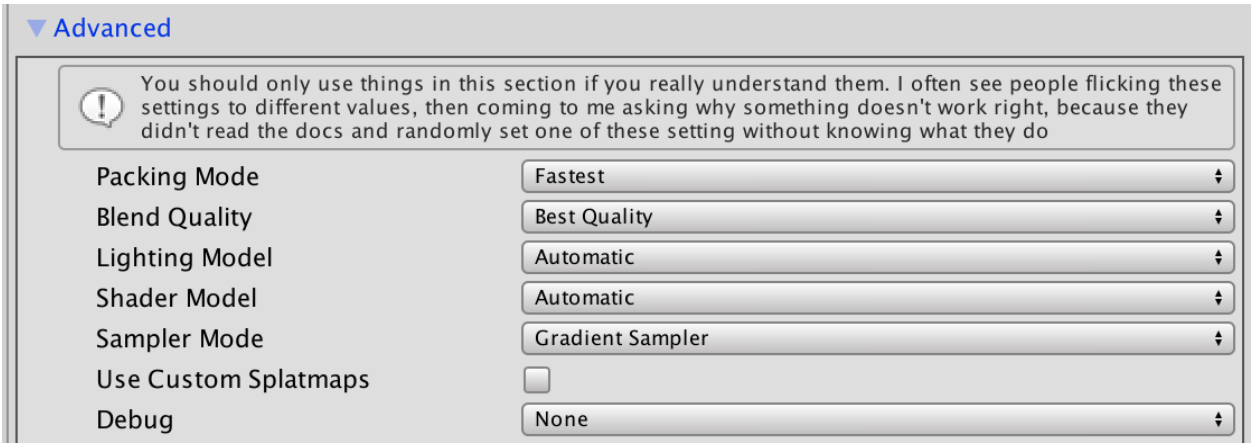
Notice that the linear blend, when far away, mixes the colors giving you a soft blend between the textures. However, close up, it's a blurry mess. Notice that the height map blend loses the soft color mix far away- this is because in a height map blend, one texture tends to win over the other. But up close, the sand appears between the rocks, and looks much better than the linear blend.

The Hybrid Height Linear blend mode gives you the best of both worlds, interpolating the height map blend to be a linear blend over the distance you specify.

- **Normal Mode**.
    - **Disabled**. This will disable the sampling of normal maps, which can be useful for non-PBR looks which don't need normals. Note that Smoothness and AO will also be disabled if you are in the fastest packing mode. Also, normals for modules outside of the Splat Mapping may still provide normals.
    - **AutoNormal**. This mode uses no normal maps for splat mapping- instead, normals are computed from the height map (which is packed into the diffuse) and requires no additional samples. It does have some artifacting when you are close to the surface, but otherwise looks surprisingly good and can save significant bandwidth on low end platforms.
    - **Regular.** This mode is the default, using traditional normal mapping for normals.
    - **Surface Gradient.** In this mode, a new surface gradient framework is used instead of traditional normal mapping. Surface Gradients allow for better blending of normal maps in certain cases, particularly when layering multiple normal maps on top of each other, such as when using noise normal or detail/distance noise. It will also change the mapping of the textures in Triplanar mode, as in Fast mode an axis flip is used to fix the normal direction. Note that in most cases of normal blending throughout MicroSplat, the result is identical or very similar.
- **Control UV Noise**. This allows you to use a small amount of noise to distort the UV coordinates of the splat map lookups. This can hide the low resolution of the splat maps, be used as a noise based transition instead of a height based one when not using height blending, and provide extra variation in the painted map.

Advanced Settings

The advanced settings are where I put features that new users tend to randomly change without understanding things. The settings here can cause blending issues, lighting issues, and require you to pack your textures differently. There are uses for them all, but if you're not trying to solve a specific issue, you should leave them alone.



- **Packing Mode**: This allows you to choose between the two texture packing modes. See the section below for more information.
- **Blend Quality**. This controls how many textures are blended for each pixel on the terrain. While many terrain shaders sample every texture that could potentially be used on a pixel, MicroSplat samples between 2 and 4. When set to "Fastest", only the two most weighted textures are sampled, however blending quality is reduced and there can be artifacts. On "Balanced", 3 textures are sampled. And at "Best Quality" the top 4 textures are sampled. This can create bad blending between textures, and is intended for terrains in a distance and other edge cases where that can be acceptable.
- **Lighting Model (BIRP/URP only)**. This allows you to override Unity's default BDRF (specular) function.
    - For Built in Render Pipeline (BIRP) Unity normally uses a GGX based BDRF which can look too shiny on glancing angles. There are two methods to eliminate this, the first is "StandardShaderNoSheen" lighting model, the second is to turn on

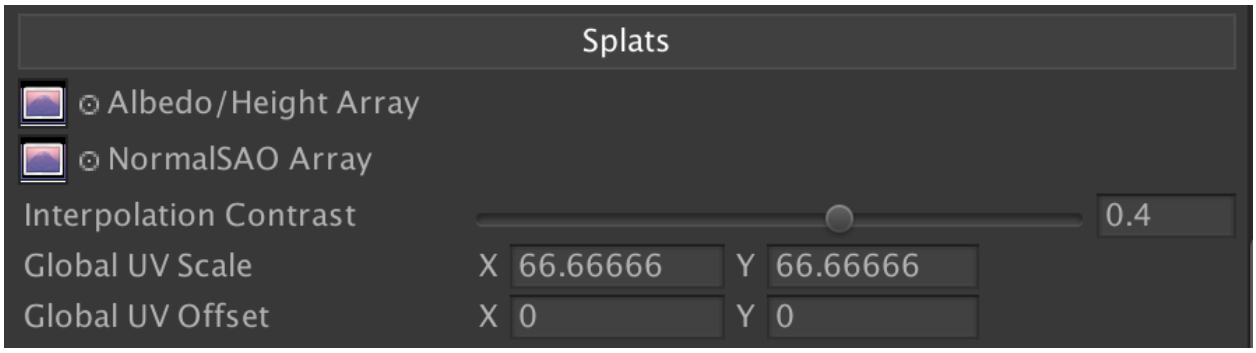
"Disable Minimum Dielectric". Additional modes are available for other BDRF functions, such as lambert shading, but note that these settings will force your terrain to be rendered as a forward renderer object when running in deferred rendering.

If set to Lambert, the shader will be compiled using the legacy Lambert lighting system from Unity4. Note that this is NOT a physically based rendering (PBR) shader. In this mode, data in the roughness texture will map to the gloss value, and the ambient occlusion will be used for specular power.

  - For URP, the SimpleLit model is available.
- **Shader Model**. By default, MicroSplat will choose the minimum shader model possible for your shader. Shader Model 3.5, or 4.6 if Tessellation is enabled. However, you can override the shader model to 4.6 or 5.0 using this control.
- **Sampler Mode**. This allows you to force the textures to be sampled using the LOD or Gradient sampler modes. Some features, such as Branch Sampling or Per Texture UVs require the sampler mode forced to LOD or Gradient. Not that the LOD sampling mode is the fastest, but breaks trilinear and anisotropic filtering. Gradient sampling tends to perform similar to default, and is usually what you want.
- **Use Custom Splatmaps**. When this option is enabled, splat maps can be supplied on the MicroSplatTerrain component instead of using the splat maps provided by the Unity Terrain system. This is most useful if you want to procedurally generate or modify texturing choices and want to avoid the hitch from basemap generation with standard Unity Terrain updates. Note that the splat maps have to have a normalized weight (totalling 1) across all splat maps.
- **Debug**. The debug section lets you view all kinds of data from the shader, such as how many samples are being performed for each pixel, what individual data from the shader such as the smoothness or normal maps look like, etc.

Shader Properties

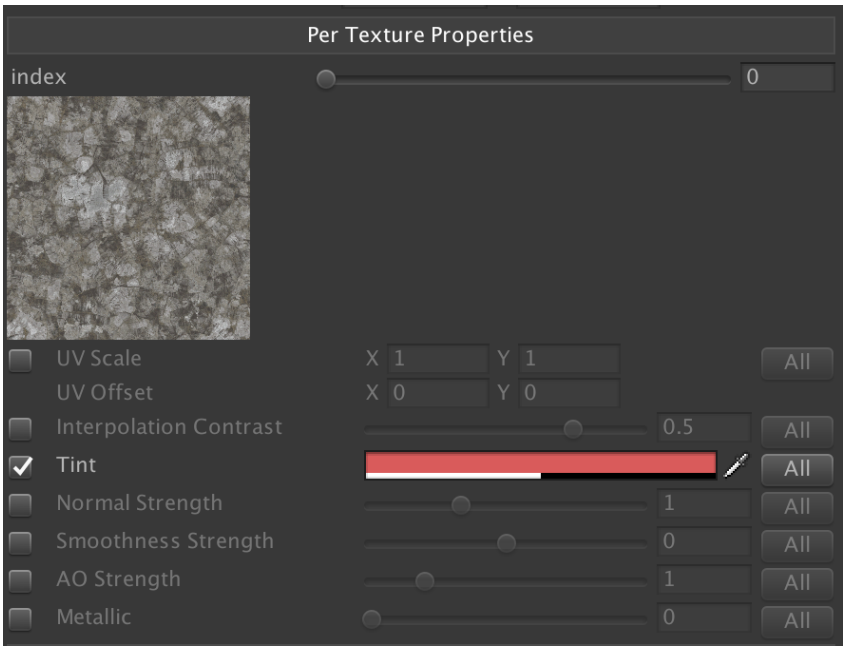
Below the compiler are various shader property sections.



This is the main splat mapping parameters. The Texture Array Config outputs our full PBR texture set into just 2 texture arrays, which you can see assigned here. If the Emissive/Metal array option is enabled, a third Texture Array is shown for the emissive/metal array.

- Interpolation Contrast
    - This controls the blend between textures when painted on the terrain. At 0, it closely mimics a Unity terrain. At 1, the blend is extremely sharp between two textures, and based on the height maps.
- Global UV Scale/Offset
    - These allow you to control the amount of tiling the textures use over the terrain, and offset the textures as well.

Per Texture Properties



The Per Texture Properties section of the shader UI lets you control various parameters of each texture in the shader. This allows you to give each texture it's own UV scale and offset, tint colors, or adjust PBR parameters like how smooth or metallic the surface is. Additional modules will add additional Per Texture Properties for their various features, and all properties have tooltips if you hover over them to explain what they do.

Next to each section of properties is a checkbox. This allows you to toggle the effect on and off. Doing so will recompile the shader with that code removed, ensuring the most optimal shader possible. Once enabled, you can use the "index" slider to choose a texture and adjust the properties. If at any time you wish to set all the textures to the same value, you can press the "All" button and your current settings will be applied to all of the terrain textures.

Many of these features should be self-evident as to what they do, however a few warrant explanation.

- MicroShadows

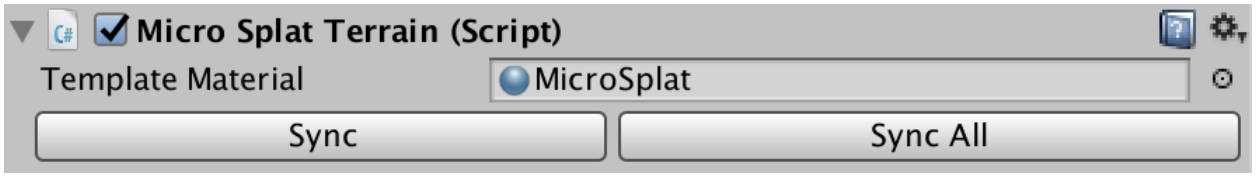
        - This is based on the micro shadow system described in the talk on Uncharted, and produces the effect of inner shadowing in the texture based on the AO.
- Fuzzy Shading
    - Fuzzy shading is a technique often used in the Unreal engine to cheaply simulate moss and velvet lighting. When enabled, the view angle is used to darken or lighten a texture, which a power function for the falloff.
- Curve Weights
    - Curve weights alters the weighting of the textures such that a curve is formed between the bilinearly interpolated splat map data. At 0.5, you will get normal blending, but at 0.01 you will get a very tight line that conforms to a spline like border between your textures. This can be very useful on paths, or creating the type of transitions found in golf courses. Since a curve is created between the points, this can look much higher res than the splat map resolution.
- Outlines
    - Outlines let you create a tint color in the blend area between the textures. Each texture can have it's own tint, and the alpha value controls how much the tint is used. A color of 0.5 will also produce no tint.

With curve weights set to 0.01, the grass area becomes smooth, like a neat path or golf course.

Debug

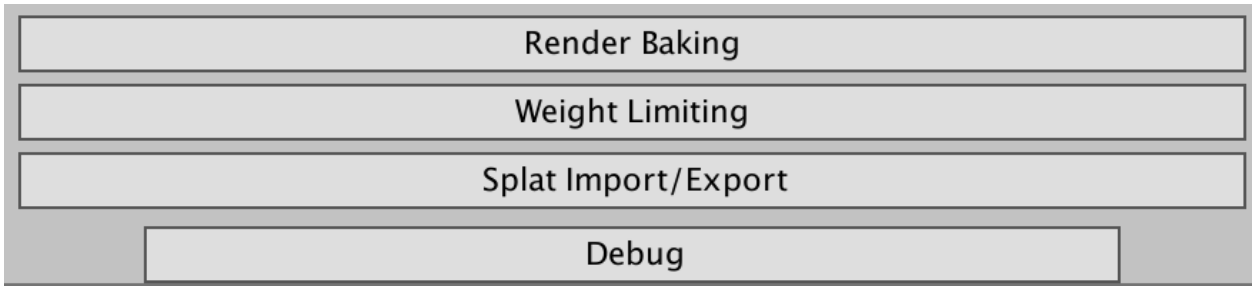
The debug section of the shader UI shows you information about the cost of your shader. While these numbers are somewhat subjective, they can be used to learn about the relative cost of each feature you enable.

MegaSplatTerrain Component



The component placed on each Terrain has a reference to the template material. You can quickly get to the material settings by double clicking on this material. When multiple terrains are setup together, a single template material is used for all of the terrains, and MicroSplat manages syncing changes from the template materials to the actual materials the terrain uses internally. If you change settings on the template material via script, you can call MicroSplatTerrain.SyncAll() to perform the same sync. Sync buttons are exposed, but you really shouldn't have to use them unless you change the template material used by terrain.

Additional modules may install additional controls into this section.



Several rollouts are available for various utilities:

Render Baking

The render baking section can be used to bake out maps from the terrain, such as the overall albedo, normal, or other PBR values.

Weight Limiting

A primary optimization of MicroSplat is controlling how many texture sets are sampled for each pixel from the terrain textures. This can be set from from 2 (fastest) to 4 (Best) with the Blend Quality setting. In most workflows, it's rare for more than a few textures to be used on a single pixel, but some external tools will output splat maps which use more textures than this. The weight limiting tool will adjust your splat maps so only the most dominant maps are used on any given pixel, preventing these types of issues. It can also be used in conjunction with the faster blend quality options to reduce errors caused by this type of optimization.

Splat Import/Export

These tools can be used to import or export splat maps from the Unity terrain. When importing splat maps, the textures will be resized to the size of the terrain data. Also, it will only import splat data for channels with textures assigned.

Debug

In normal operation you should never have to look into this section, but occasionally a user will manage to unhook various data which MicroSplat generates and manages, so these references are exposed here. These include the PropData object, where MicroSplat stores information about per-texture property values, and the keywords object, where MicroSplat stores data about the features you want compiled into your shader.

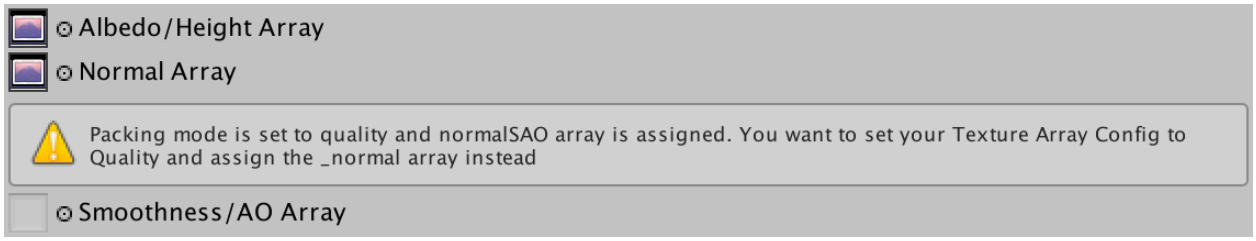
Packing Modes

On the Texture Array Config and shader editor there are choices for Packing Mode. This allows you to choose between two packing formats for texture data, with the tradeoff being between speed and quality. In the default (fastest) mode, the texture data is packed into two Texture Arrays; one for Diffuse and height data, and another for Normal, Smoothness, and AO data. When using the “quality” setting, the normal maps are put into their own array, and the smoothness/ao data are put into another array.

When using texture compression, this can increase the quality of the normal, smoothness, and AO maps over the more compact packing mode. This is because texture compression considers RGB values as one compressible thing, and uses knowledge about how our eyes see to bias that compression. So having data in all 3 channels creates more compression artifacts than if you just put data in the G channel. And Alpha is usually compressed separately. How much this matters will really depend on the type of textures you are using, but in most cases I find that the quality difference is relatively minor, and running in the fastest mode is best.

Note that the packing mode on the shader and Texture Array Config must match that in the shader. Once this is set, the Texture Array Config will output texture arrays into the

MicroSplatData directory with the post fix names of _normal and _smoothAO array instead of the traditional _normalSAO one.



These must be assigned to the material under the splats section of the shader into the matching texture array slots. Note the warning is show here because the editor has found the _NormalSAO extension on the array assigned to the Normal map. This texture array still has the smoothness and AO settings packed in with the normal, providing no increase in quality. In this case, the _normal array needs to be assigned instead, and the _smoothAO array assigned to the slot below it.

Additional Modules

Additional modules can be downloaded to expand the feature set of MicroSplat. Once downloaded and installed from the asset store, new options will appear in the Shader Generator section. Documentation for all modules is included with this package in case you have any questions about the workflow. This unique model allows you to pay for only the features your game needs, instead of paying for one large expensive package.

Global Properties

Often you might want to control aspects of MicroSplat from an external control system, like a weather manager such as Enviro. When a “G” button is located next to a property, you can toggle it to be driven by a global shader property. When these are enabled, the G will turn yellow and the property will no longer be editable.

You can then set these values globally via code, using the Shader.SetGlobal function, something like:

Shader.SetGlobalVector("_Global_WetnessParams", new Vector2(minWetness, maxWetness));

A list of all potentially global values:

_Global_WetnessParams	Vector2	Minimum and maximum wetness in the scene. Zero to one values expected.
_Global_PuddleParams	float	Maximum puddles in the scene. Zero to one values expected.
_Global_StreamMax	float	Maximum height of streams. Zero to one values expected.
_Global_RainIntensity	float	Rain drop intensity. Zero to one values expected.
_Global_SnowLevel	float	Overall amount of snow. Zero to one values expected.
_Global_SnowMinMaxHeight	Vector2	Begin and end height of snow in scene in meters.
_Global_WindParticulateRotation	float	Direction of wind particulate flow in radians. Should not be animated at runtime.
_Global_WindParticulateStrength	float	Strength of wind particulate effect. Zero to one values expected.
_Global_SnowParticulateStrength	float	Strength of snow particulate effect. Zero to one values expected.

Vegetation Studio Integration

When Vegetation Studio is installed, a few extra options are exposed. “Vegetation Studio Grass Mask” can be used to add foliage into the terrain texturing at distances beyond which it is feasible to draw actual foliage.

“Vegetation Studio Shadow Map” can be enabled to add raycast shadows for distant trees. This allows tree’s and other objects well past Unity’s shadow bounds to still cast shadows onto the terrain that correctly move and elongate with the sun position and angle.

Finally, MicroSplat can render out a tint map, which allows the bottom of the grass to be tinted towards the terrain texture using an appropriate shader on your vegetation.

Please consult the Vegetation Studio Documentation for more information on these features.

Working with Multiple Terrains

First, make sure all your terrains have a unique name. MicroSplat uses the terrain name as a prefix to generate various resources it needs for each terrain, so if multiple terrains have the same name, they will overwrite each other.

When working with multiple terrains, you can setup all the terrains as one “system” by selecting them all and doing the conversion. This will create one MicroSplatData directory for all of their data, with a single shared material, shader and per-texture property object used by all of the terrains. Modules which need to generate additional textures or data will create these based on the terrain name and place them in this same directory.

If you wish to have multiple terrains that do not share settings, it is suggested to put each of the TerrainData assets Unity creates into it’s own folder and set them up one at a time.

This will give them each their own MicroSplatData directory, with their own shader, material, and per-texture property data object.

Note that you should avoid renaming things in the MicroSplatData directory, as this can cause various issues when objects need to find each other, often causing MicroSplat to regenerate the data it needs from scratch, or potentially finding the wrong data. Also note that textures, like global maps, can be overridden on the MicroSplatTerrain object for each terrain- so you could have one MicroSplat system which is used across multiple terrains, but each terrain can have it's global maps.

Origin Shift

MicroSplat supports origin shifting for large world support. You can enable this in the advanced section of the shader generation section. Once enabled, you will need to set the global matrix in your origin shifting code:

```
Shader.SetGlobalMatrix("_GlobalOriginMTX", mtx);
```

You may also need to texture your terrain using WorldSpace UV mode.

Copying Scenes

When copying scenes to make variations, it's important to realize how both terrain data and MicroSplat serialize and reference data. The terrain itself usually references a TerrainData object on disk, though this can sometimes be stored in the scene itself. Unity references other files via metafile information, which can be stored in .meta files or baked into the objects themselves. So when you copy something like the MicroSplatData folder to a new location, those files will have the same meta data, and Unity will detect this issue and assign the new files new meta information. However, any references to old files will be kept. So for instance, the new material will still point to the old shader, and the terrain component will still point to the old

material, and any other references it has to the old data. So if you are going to duplicate a scene, it is important that you duplicate all of the data, and point all references to the new data so as not to edit the old data by mistake.

Additional Information

Are you interested in creating assets for sale that use this shader system? Do you make an asset which could use a better shader in your demo scene? If so, please contact me and I can help you do this with a special version for inclusion into your asset.

Are you a shader developer who wants to extend this system with new features, even selling an extension module on the store yourself? If so, please contact me and I can help you do this.

Removing MicroSplat

MicroSplat edits several settings on the terrain which will need to be fixed if you remove it:

- Depending on Unity version, you may need to assign the appropriate terrain material back to the terrain.
- The base map distance will likely need to be set to something higher for the Unity shader
- MicroSplat may have needed to add new layer files to the terrain, which would be located in your MicroSplatData directory. It's best to move these somewhere before deleting the directory.