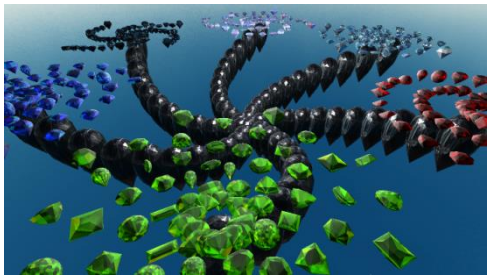
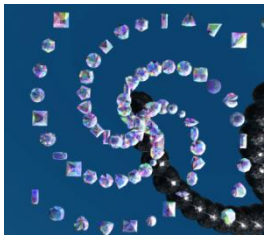
ADVANCED GEMSTONE SHADERS

INTRODUCTION:



I am fascinated from crystals and minerals and gemstones. As a game developer, I always look for solutions to simulate such precious stones. When most people hear the word gem, they automatically think about diamonds, sapphires or rubies as they are most known to the mayor population. However there are way more different kinds of stones like opals or moonstones which are usually cabochons and are very difficult to simulate with shaders. I have checked already existing gemstone packages but was not satisfied with what I got.

ABOUT THIS PACKAGE:



Occasionally I have the need to work on shaders and suddenly 12 hours have passed. This package contains shaders I have made to simulate gemstones and will be updated whenever new shaders for gems are implemented. It also contains 50+ gemstone models of common shapes found in the jewelry business and a simple pattern generator to arrange these objects.

THE SHADERS:

Shaders of this package sometimes work different than standard shaders you know to archive such effects. However one common thing you will notice is the high usage of cube maps. They are indeed important to simulate the reflections and effects which occur in gemstones.

The aim for these shaders is to make it possible to create many different materials with a few textures as in my home environment, artists are very rare. Also keep in mind that some shaders require a reflection probe in order to look nice.

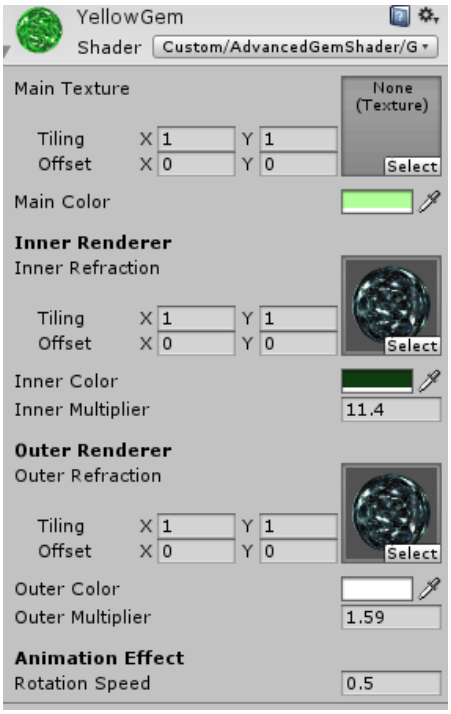
The best thing is really to just play around with the values and parameters. Often fancy looking materials were created accidentally by just experimenting with parameters.

Another thing is that there are so many different kinds of gemstones that this package will never be completed and there is always something new to discover and improve. Also when new technologies emerge which may allow the simulation of inclusions.

SHADER 1: GEMSTONE SHADER SIMPLE

This is the first shader and the most simple one. You have full control over the visual appearance of the gemstone. However it does not look fully realistic but it does not need reflection probes.

This effect is acquired by rendering the interior and the outer parts separate. The most important parameters are the refraction and reflection textures which are simple cube maps. In the hints section are some tips about how to create them fast.



Parameters:

- *Main Texture: standard main Texture*
- *Main Color: standard main Color*

Inner Renderer:

- *Inner Refraction: Inner reflection Cubemap*
- *Inner Color: Cubemap Color Multiplier*
- *Inner Multiplier: Power Multiplier*

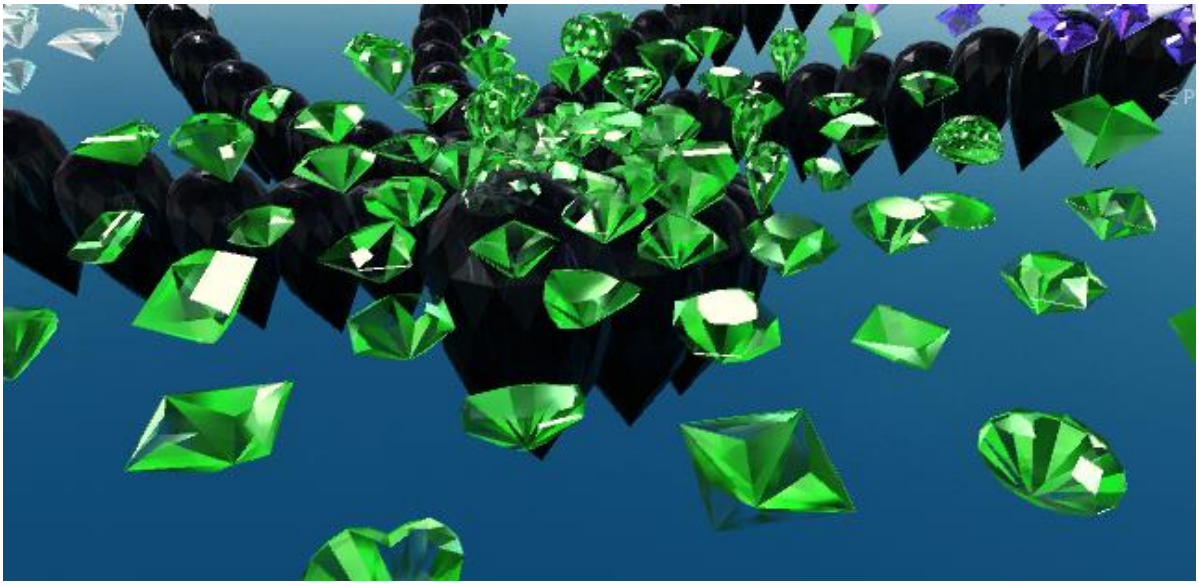
Outer Renderer:

- *Same parameters as Inner renderer. Renders the outside of the gemstone*

Animation Effect:

- *Rotation Speed: Lets the cubemap rotate to give those gemstones magic like animations.*

Usage: This shader is good for mobile applications or if the gemstone should stay unaffected from external influences. It is also possible to create unnatural gemstones where the sparkle color is black which is impossible in real life. Check out the "Curse" Material



SHADER 2: GEMSTONE SHADER REALISTIC

This is the second shader and attempts to simulate gemstones as realistic as possible**. This shader requires a reflection probe.** If you don't want to use it, use the simple shader instead as It looks terrible without reflections.

This effect is also acquired by rendering the interior and the outer parts separate. The refraction and reflection textures are just multipliers for the probe reflection.

Parameters:

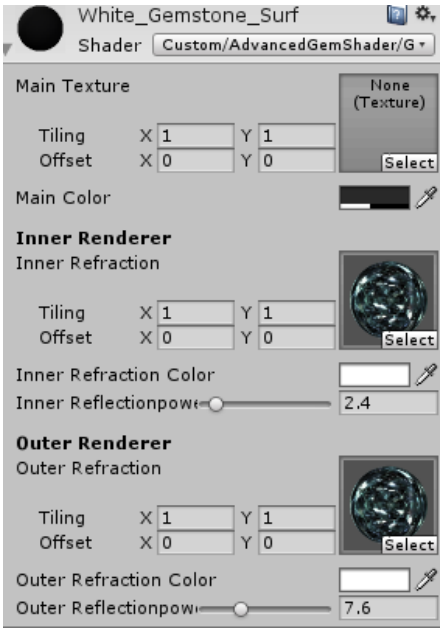
- *Main Texture: The Main Texture.*
- *Main Color: Color Multiplier*

Inner Renderer:

- *Inner Refraction: Inner reflection Cubemap*
- *Inner Color: Cubemap Color Multiplier*
- *Inner Reflection Power: Influence Factor*

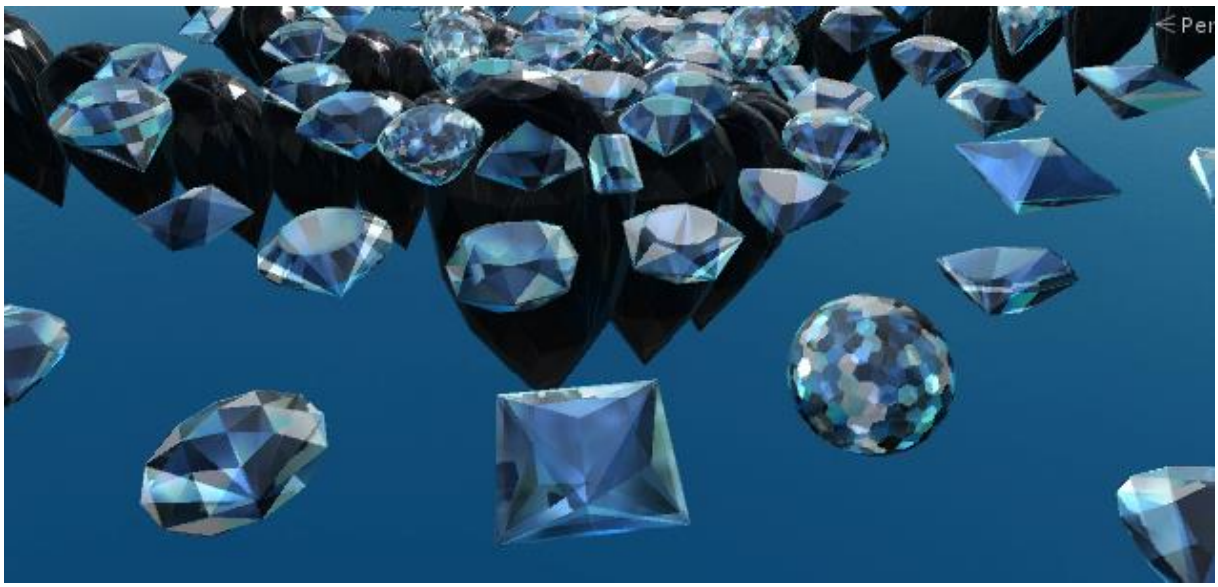
Outer Renderer:

- *Same parameters as Inner renderer. Renders the outside of the gemstone*



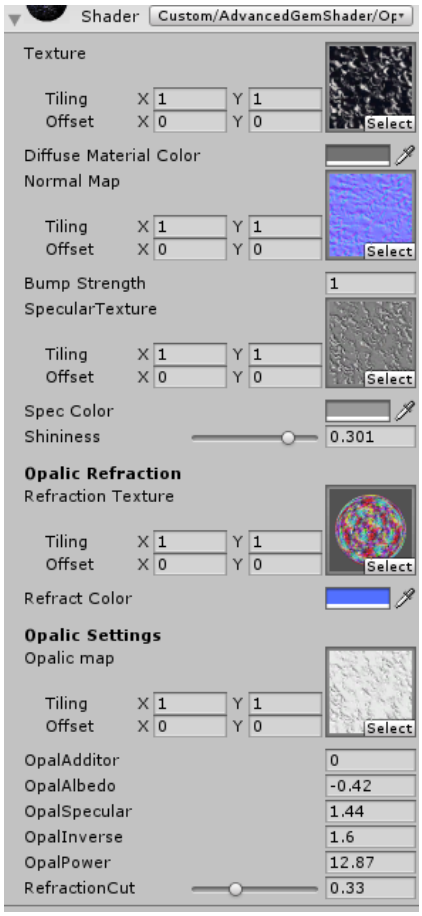
Usage: This shader is good for realistic applications. Keep in mind that the gemstone will be just dark in environments without light and reflection like in real life.

These 2 Shaders handle normal gemstones like diamonds with all their colors. Although they are not 100% realistic as we would need ray tracing which doesn't work in real time at the moment. However with the upcoming OpenGL update Vulcan, better results could be possible.



SHADER 3: OPALIC NORMAL/REFRACTIVE

In real life a special kind of gemstone exists which is completely different than common gemstones which are called opals. Opals are a special form of silica which is amorphous. Due to water inclusions, opals have sparkles in all colors. The right image is a real example of such stone. Currently rendering one would require some kind of 3D-Texture map.(image from Wikipedia)



This shader maybe comes close to 30-40% of a real opal as graphic engines have to be able to render inclusion without altering the actual geometry. This may be possible with procedural 3D Textures in the future. This shader uses the standardspecular shader as base

Parameters:

- *Texture: The Main Texture.*
- *Diffuse Mat Color: Color Multiplier*
- *Normal Map: Standard normal map*
- *Bump Strength: Normal power*
- *Specular Texture: Normal specular texture*
- *Spec Color: Color Multiplier*
- *Shininess: Shininess for Reflection Probe*

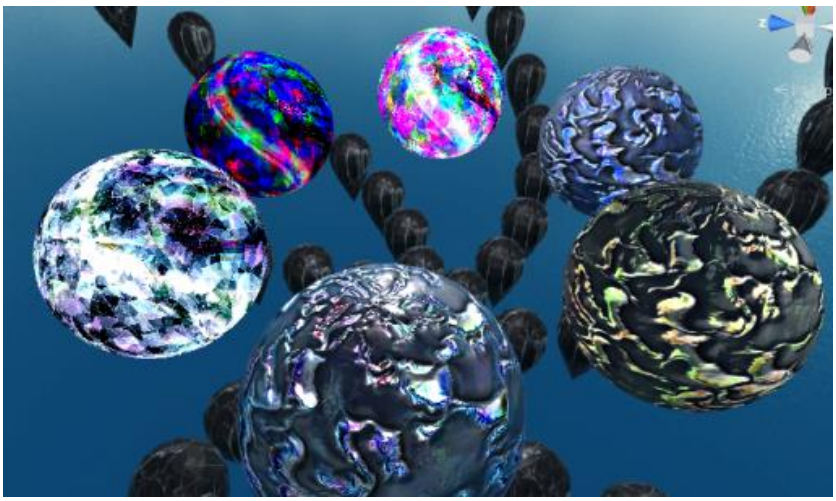
Opalic Refraction:

- *Refraction Texture: Reflection texture of "opalic" Sections*
- *Refraction Color: Color Multiplier*

Opalic Settings

This is a texture with an alpha channel. Alpha describes where "opalic" spots are. Color of this texture does not matter.

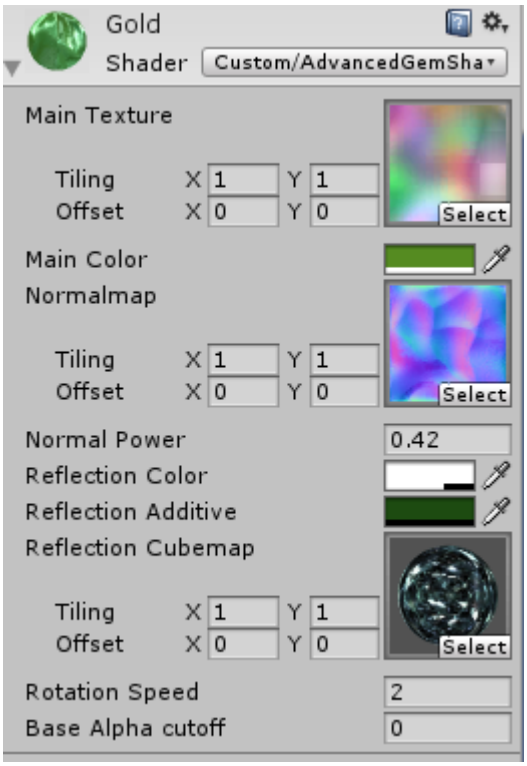
- *OpalAdditor: Add value to the opalic value*
- *OpalicAlbedo: Influence to the albedo map*
- *OpalicSpecular: Influence to specular map*
- *OpalInverse: Multiplicator of the opalic map.*
- *OpalicPower: Power for the opalic map. Values with low alpha stay low while values with high get higher*
- *RefractionCut: Final multiplicator to the opalic value*



SHADER 5: METALLIC:

Sometimes you face the problem with simulating metallic material in 2D games. Usually you can use the standard shader and set the metallic value to maximum and throw in a reflection probe. However in 2D games, the metallic properties are not noticeable because the camera angle does not change.

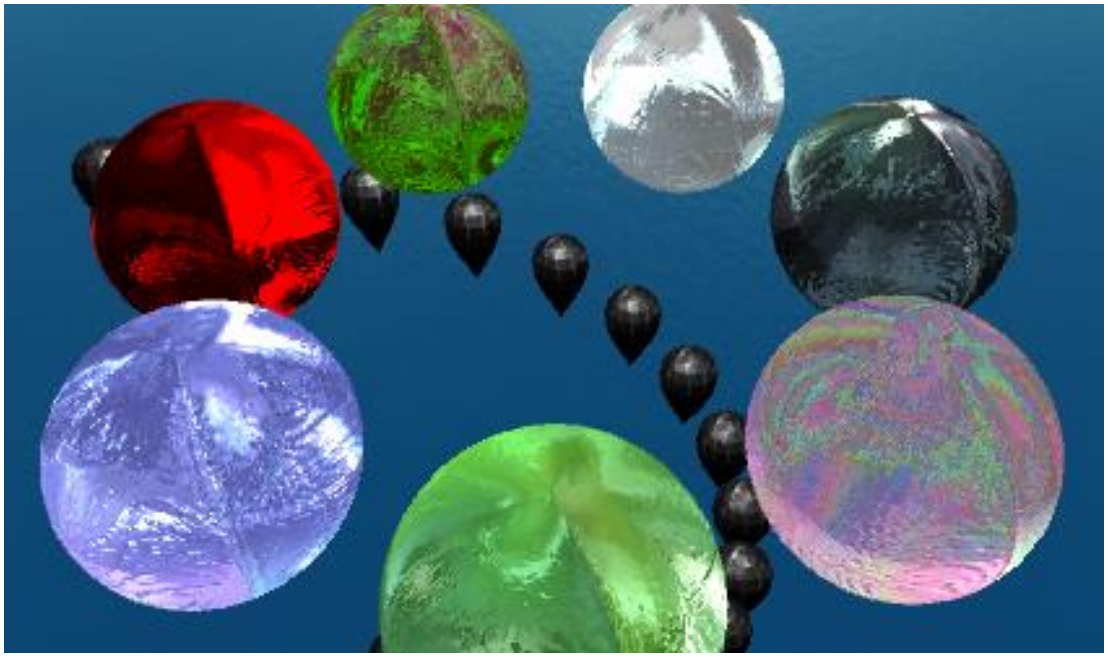
This shader aims to simulate metallic structures without reflection probes and perspective cameras.



Parameters:

- *Main Texture: Standard Main Texture*
- *Main Color: The Main Color.*
- *Normal Map: standard normal map*
- *Normal Power: normal map power*
- *Reflection Color: reflection color*
- *Reflection Additive: fixed color added to the reflection*
- *Reflection Cubemap: Reflection map*
- *Rotation Speed: Reflection rotates so it can have animations.*
- *Alpha cutoff: supports cutoff to simulate fences, grids etc.*

This shader has 2 variants, one with cutoff and one without. If your stuff don't need transparency, use the opaque version as opaque shaders always have a better performance than transparent shader.



SHADER 6: ALEXANDRITE

Alexandrite is an gemstone with very special properties. Its appearance depends on the color of the light. For example such gem is yellow in sunlight and blue in red light. Their internal structure changes the wavelength of incoming light. It uses the same parameters of the realistic gemstone shader and requires an reflection probe.

Parameters:

- ***Red/Green/Blue Color:***

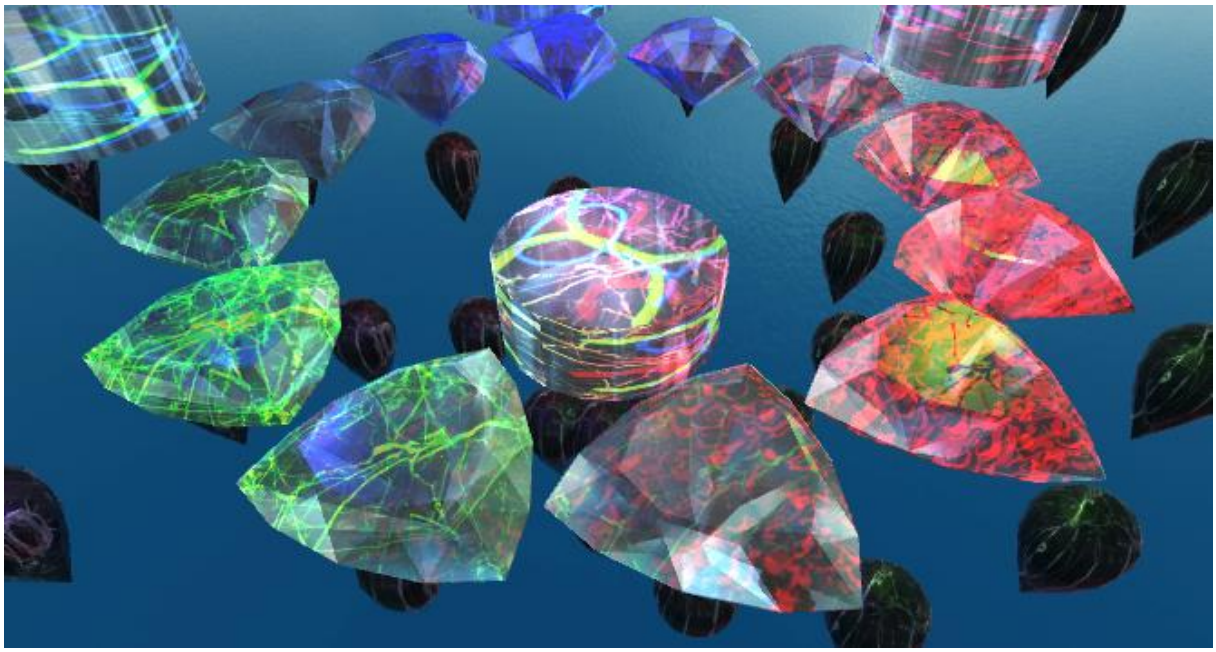
The color it should emit when colored light hits this material.

- ***Red/Green/Blue Influence:***

Power of influence: 0 means that colored light of the specific color has no influence.

- ***Red/Green/Blue Texture***

Texture influences the specific color emitted from the gemstone. It is also possible to make material which reveals hidden stuff or information when exposed to colored light. The sample shows a material where runes are visible under certain light. This effect is similar to fluorescence where materials emit yellow light when exposed to UV radiation

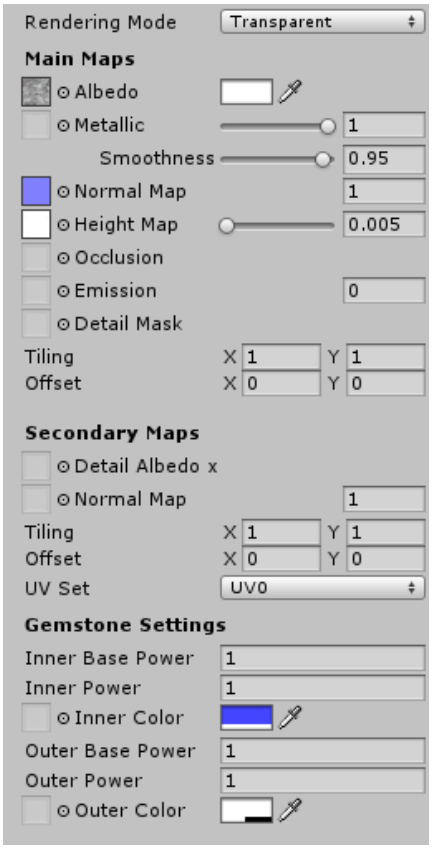


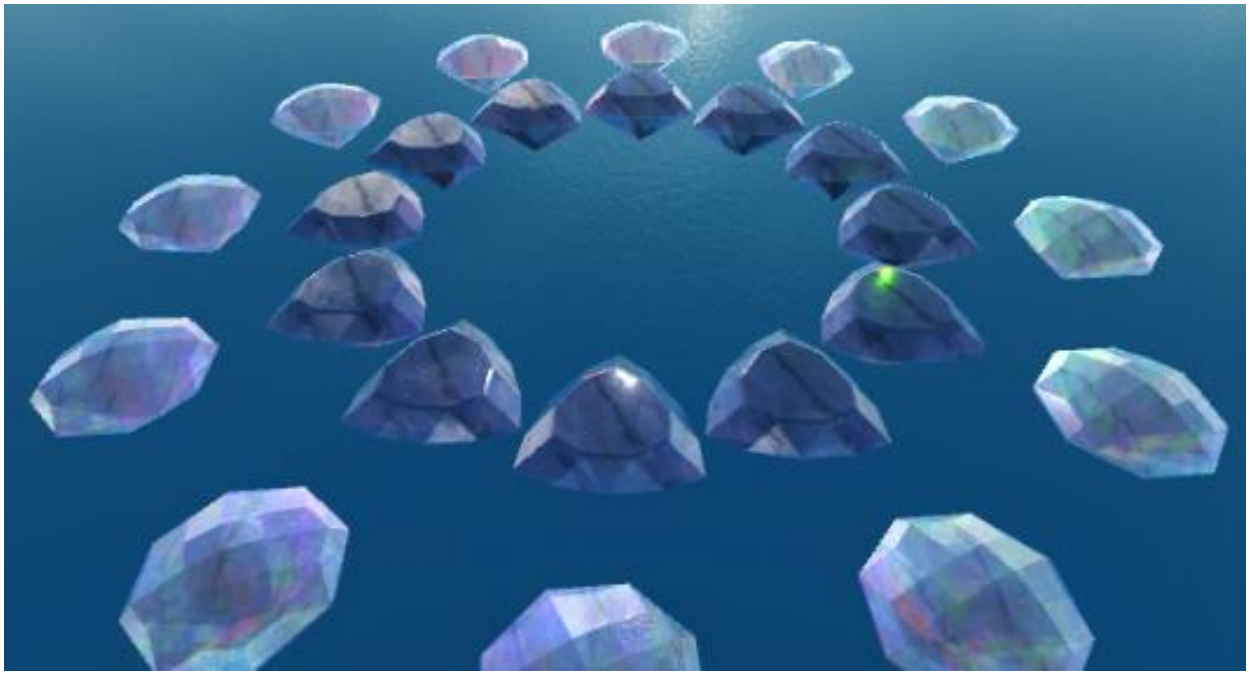
SHADER 7/8: STANDARD VERSIONS:

The standard gemstone shaders come in two variants which are the normal standard and standard specular versions. They are extended versions of the original standard shaders so they become gemstone like properties.

The main difference is only the addition of the Gemstone Settings. Parameters are also split into inner section and outer sections and are simple color multipliers.

Keep in mind that this shader only makes sense if the rendering mode is not set to opaque.





SHADER 9: COMPLEX

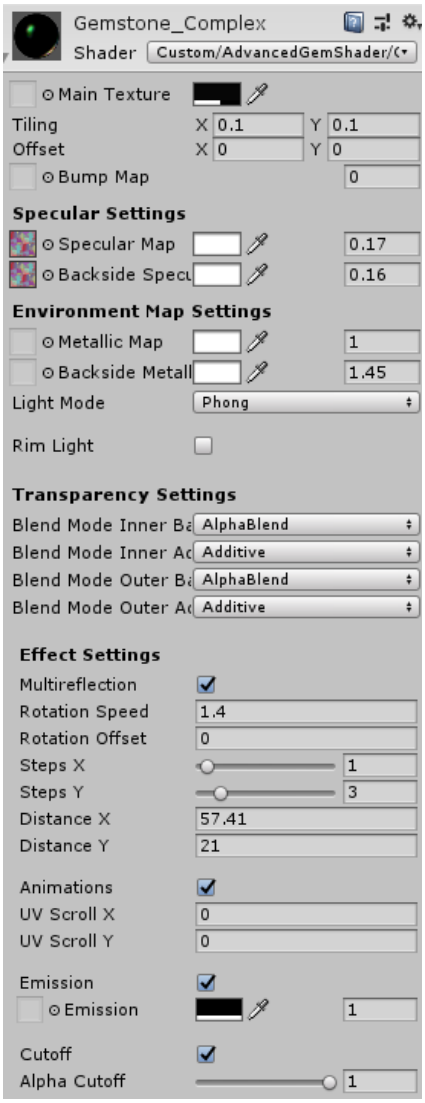
This shader allows more control over the rendering calculations unlike surface shaders which are somewhat restrictive.

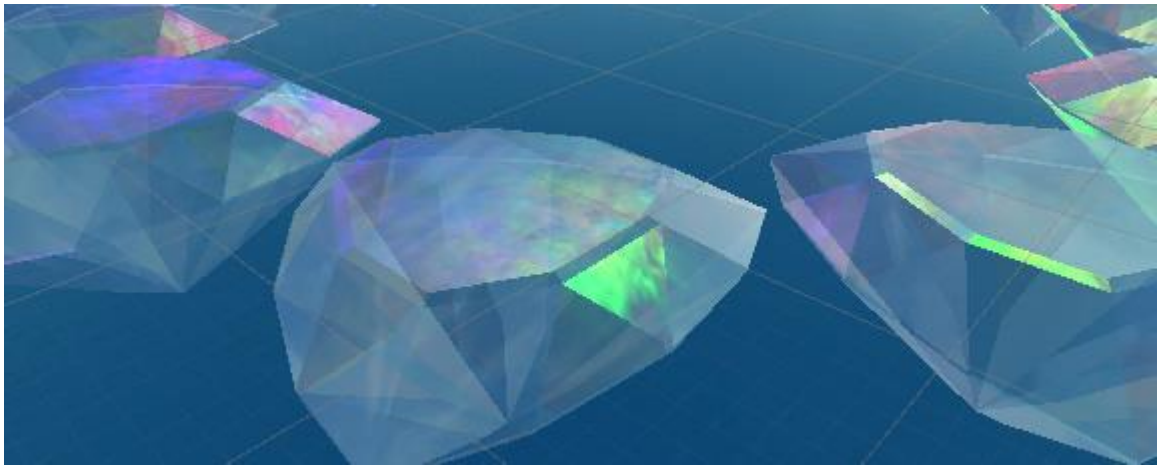
It is a custom implementation with a custom lighting calculation. It is possible to change the blend modes of the individual render passes and it also supports rim light. The rim light also includes a simple gradient generator for the required ramp textures.

Available light modes are lambert, phong and blinn-phong. The base settings in the specular and environment settings are separate for backside and front side.

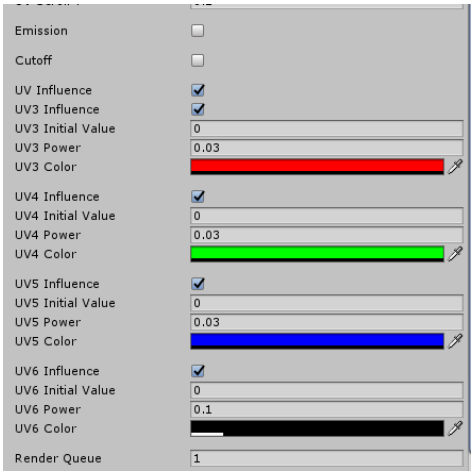
The multi reflection settings allow the implementation of kaleidoscopic reflection. Cubemap reflection is applied multiple times dependent on Steps X and Steps Y. The position for each reflection is defined by the Distance X and Distance Y parameters. Rotation Speed is for animation and the Rotation Offset is another positioning parameter for reflections.

The UV Scroll X/Y can be set in order to let the main texture scrolling. Emission lets the material glow similar to the emission value from the standard shader. Cutoff uses the alpha value from the main texture to discard pixels similar to the standard shader.





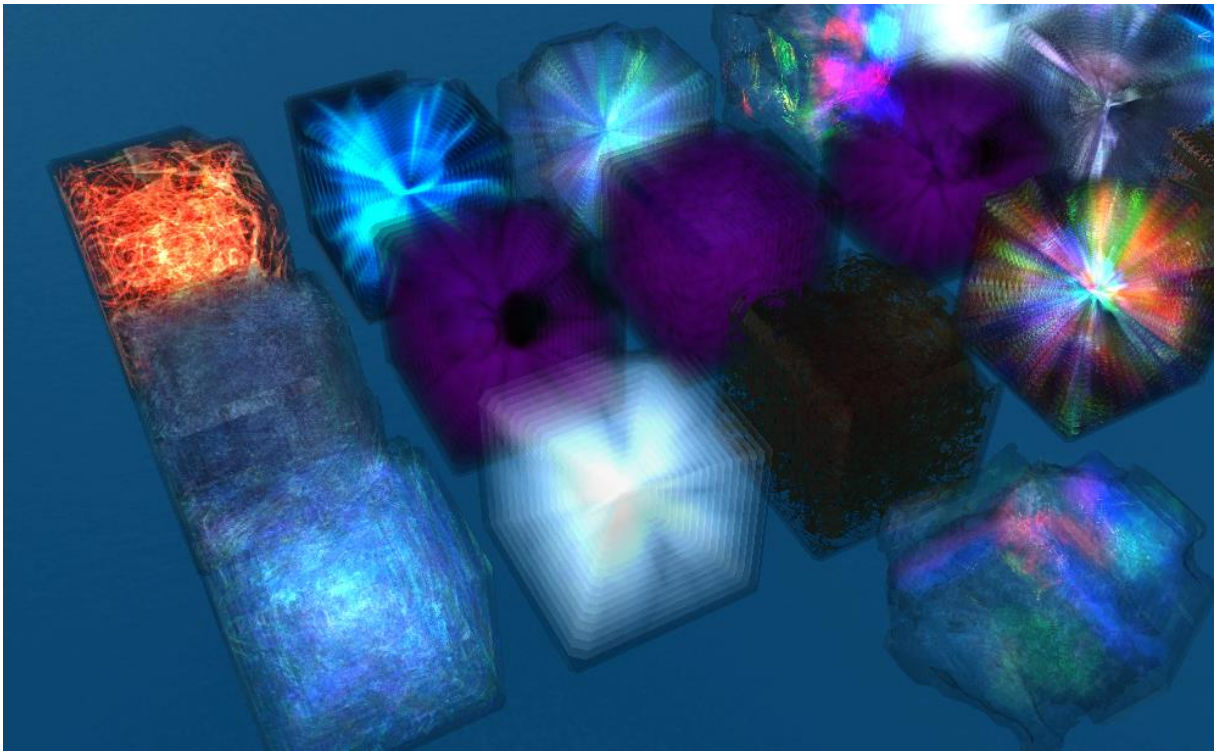
The gemstone complex shader now also has the option to be influenced by additional UV channels. Currently supported UV channels are UV3, UV4, UV5 and UV6. The main usage for these UV channels is related to procedural mesh modification as it is used by the Voxel Generator. Every channel multiplies the UV value into the material and can be tweaked with the according Initial Value, Power and Color modifier.



In order to see results, the mesh must contain those additional UV values. Otherwise the result is just an invisible material as every additional UV coordinate is 0 if not explicitly defined.

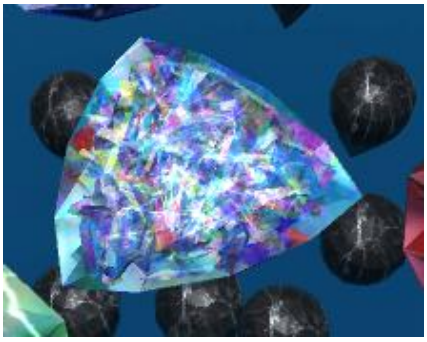
Also it is now possible to modify the Render Queue just like in the Standard shader.

The sample below shows the application of additional UV channels which were created using the Voxel Generator. **More detailed information is in the section related to Asset Extension.**



INCLUSION GENERATOR:

The gemstones in this package are actually absolute perfect gemstones. However such gemstones are very unnatural because most gemstones have errors in their crystalline structure. The inclusion generator attempts to create such "errors" by injecting meshes into the original gemstone mesh. It is almost possible to create real opalic behavior if the density of such inclusions is high enough.



HOW TO USE:

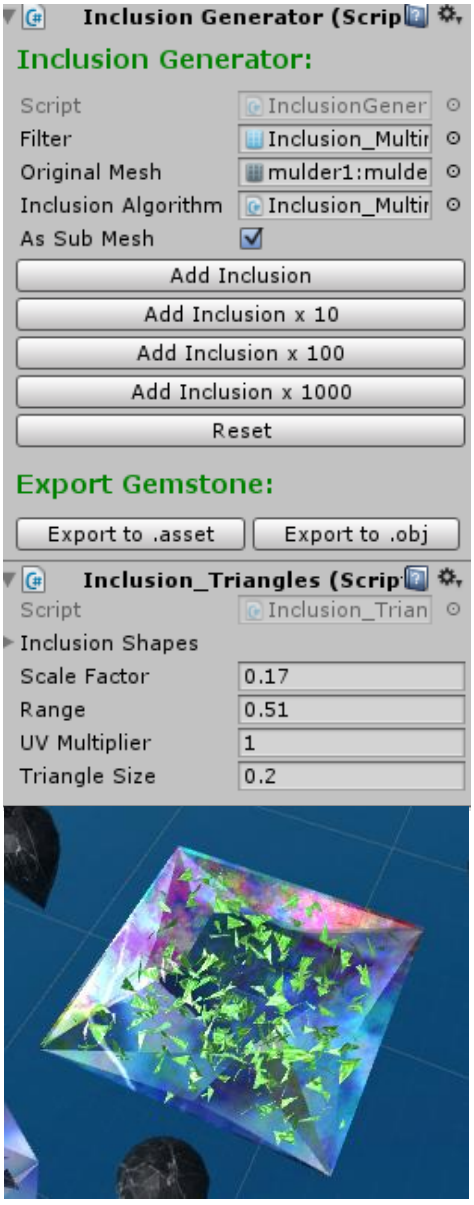
The Inclusion Generator has its own subfolder within the Advanced Gem Shader folder (Could be an own asset). Either simply drag in one of those prefabs or attach the Inclusion Generator to a game object.

The generator itself requires a MeshFilter component attached to it and an original mesh which can be any convex shape which is saved in the asset database. This shape is used to check if the injected inclusion is really inside the gemstone and not outside.

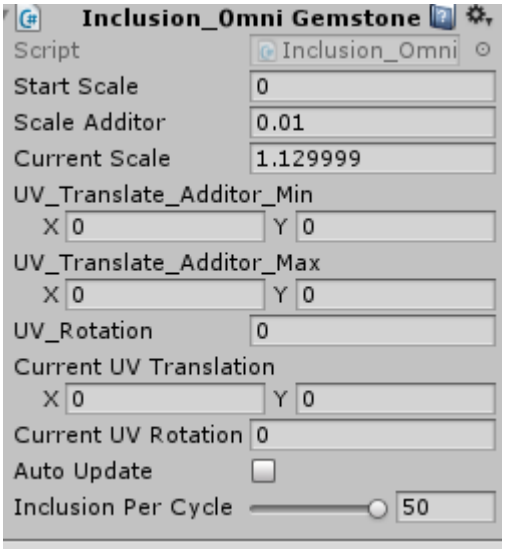
The generator also needs an inclusion algorithm which is also a component attached to the game object. It is recommended to keep the parameters very small so the inclusion is really inside the gemstone.

Whenever you hit one of the "Add Inclusion" buttons, the inclusion algorithm creates a mesh and injects it into the gemstone. However this process can fail if any vertex of the inclusion is outside of the gemstone or the maximum vertex count is reached.

Now it is also possible to separate the Inclusions from the main Gemstone. It will become a sub mesh so the final gemstone consists of two sub meshes. The resulting gemstone can therefore consist of multiple materials: One for the gemstone and one for the inclusions.



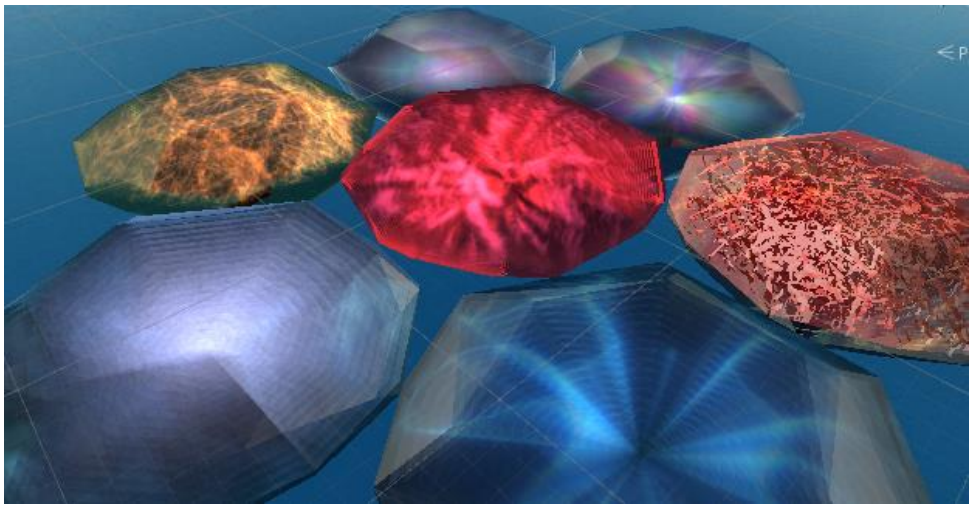
OMNIGEMSTONES:



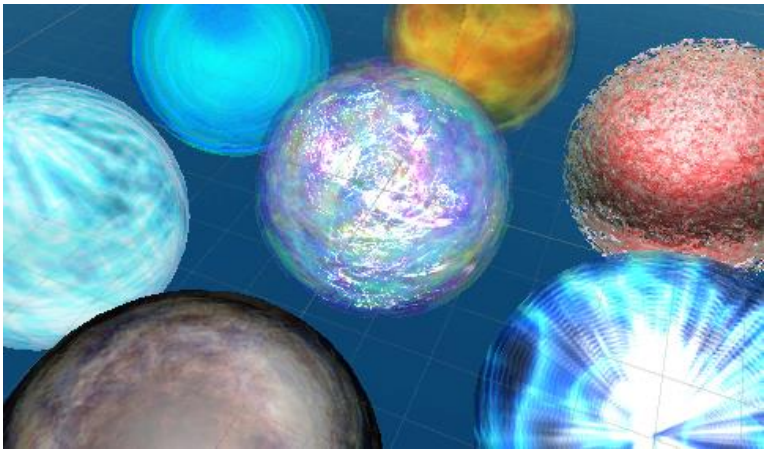
OmniGemstones can be created using the Inclusion Generator. What it does is simply creating inclusions of the same shape as the gemstone itself but with a smaller scale than the previous inclusion. If combined with transparent shaders, gemstones with very unusual properties are possible.

The start scale is the initial scale of the inclusion and the scale additor determines the growth of each layer. The current scale is simply to display the current scale. It is also possible to manipulate the UV values of each layer.

You can also see the results immediately when auto update is checked but the maximum amount of layers are 50 to prevent slowdowns of unity. It also ignores inclusion check to increase speed. If you want more than 50, hit the normal create inclusion buttons to add layers. The major effect is that 2D textures now apply a 3D effect when the material is half transparent. The image below shows such objects which exhibit an interior 3D effect.



Another sample shows the usage of the built in sphere from Unity which is transformed into an OmniGemstone. The material for these spheres uses the gemstone complex shader in order to create amazing results.



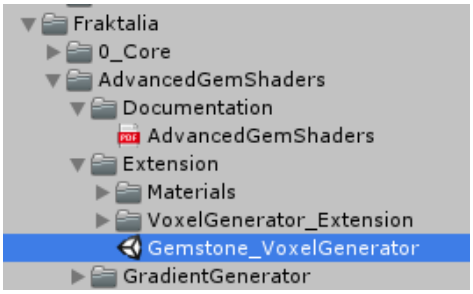
EXTENDING:

It is possible to create your own Inclusion algorithms by simply deriving from the **Inclusion_Base** class. The class itself has 4 virtual functions which are:

- **Preview():** For drawing debug stuff and Gizmos
- **CreateInclusion():** Is called when an inclusion should be created. Returns a mesh.
- **InclusionInvalid():** Is called when the mesh is not completely inside the gemstone.
- **Reset():** Called when hitting the “Reset Button”.

ASSET EXTENSION: VOXEL GENERATOR

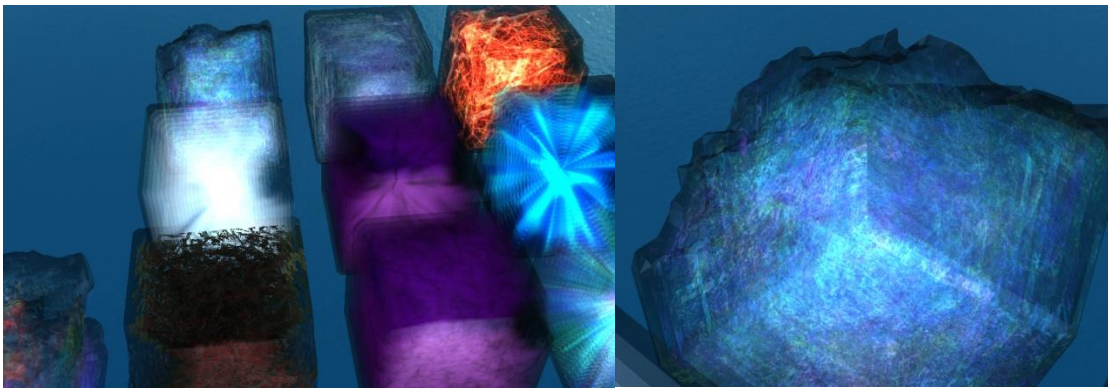
This asset contains an extension package which contains samples using the Voxel Generator asset. The extension content is imported by clicking on the Gemstone_VoxelGenerator file located inside the extension folder. The unpacked content contains a sample scene, prefabs and mesh samples required in order to create objects like in the images below. The results are fancy, sculptable blocks with volumetric interior like in the images below.



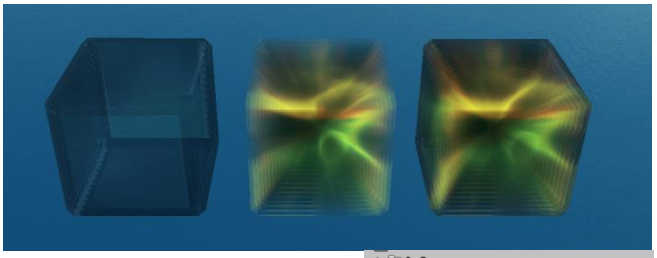
The extension content requires the Voxel Generator asset to be included into the project .

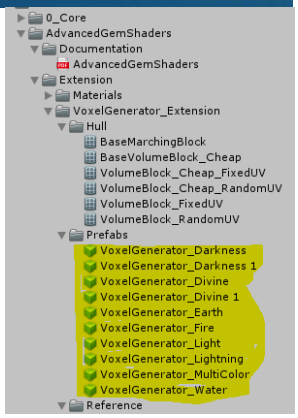
https://assetstore.unity.com/packages/tools/terrain/voxel-generator-162883

If the Voxel Generator asset is missing, only the preview content (exported voxel mesh) is shown as sculptable content cannot be generated without the voxel engine.



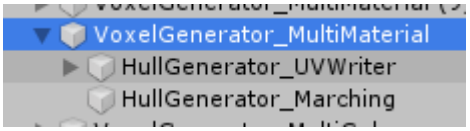
The volumetric blocks are mesh objects generated by the voxel generator. The left block in the right image shows an exported mesh generated by the VoxelGenerator and represents the outer surface. It uses the gemstone complex shader as it has normal gemstone properties. The block in the center is the volumetric interior and is an OmniGemstone, created and exported by the inclusion generator. The base mesh for the inclusion generator is the mesh for the first block. The right block shows the result when the outer hull and the volume block are combined.





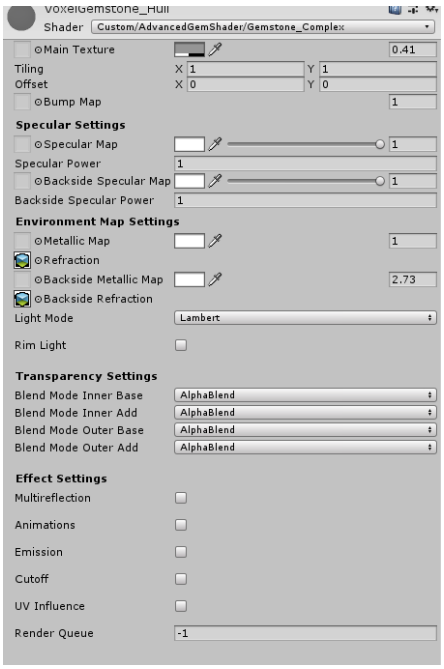
The easiest way to add such blocks into the scene is by just using those prefabs. If one is dragged into the scene, the VoxelGenerator will immediately generate the result. The block can then be sculpted with the VoxelGenerator ingame or in editor.

Inside the scene, the Prefab contains the VoxelGenerator as the root object and two hull generators which handle the visualization. The first hull generator is the UVWriter which only writes into the UV coordinates of an existing reference mesh. The reference mesh is one of the sample VolumeBlocks mesh objects. The second hull generator is then just marching cubes which uses the gemstone shader for the outer hull and generates the outer surface.

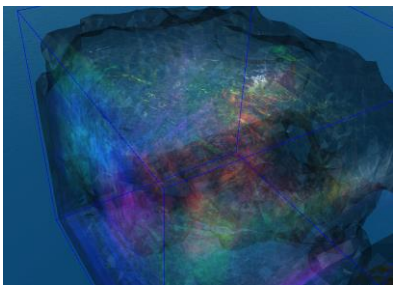


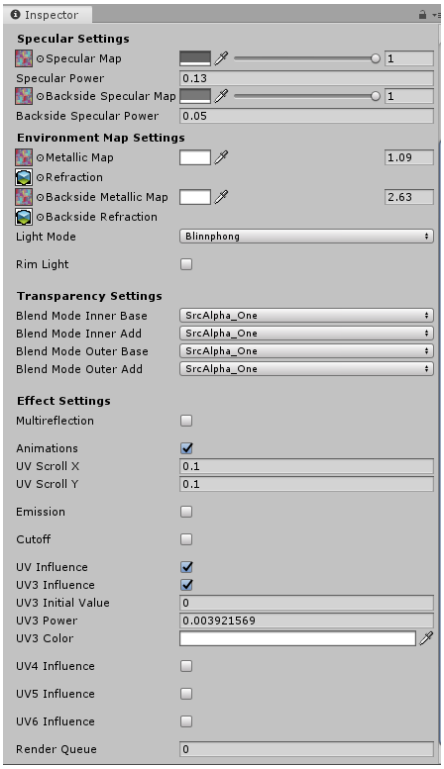
In the sample, both objects use the gemstone complex shader as it provides the maximum amount of settings including the UV influence.

The first shader is a simple gemstone material and can have any customized appearance as desired. The only important setting is the render queue which defines if the hull should be rendered before or after the volumetric interior. When set to -1, it is drawn behind the volume interior.



The material for the volumetric interior can have any material property but must have UV influence enabled. Otherwise the complete mesh is always visible like a normal OmniGemstone. It is also possible to enable all UV coordinates for example using UV3, UV4, UV5 and UV6 for each color channel: Red, Green, Blue and Alpha. However this requires that the UVWriter hull generator also is set up correctly to write into those channels. The VoxelGenerator_Multicolor is a sample setup which does this and allows to sculpt material including each color channel.



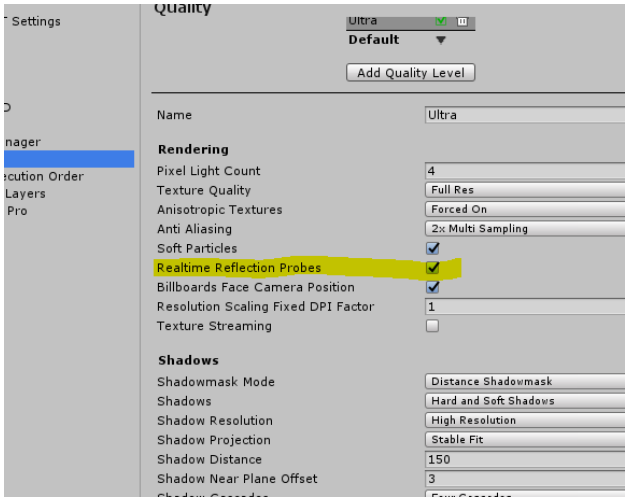


MOBILE SUPPORT

All shaders are compatible on modern mobile devices, Android and IOS. Older devices may not be compatible if the integrated GPU does not support reflection probes.

If the gemstones are not fancy in a mobile build, Check the quality settings and make sure that real time reflection probe is flagged.

Gemstones with Inclusions are also supported if the device can handle the increased vertex count which is no issue on modern devices.



LAST NOTES:

If you have any questions, suggestions, bug reporting don't hesitate to contact me. If you are going to sell a game which uses this asset, inform me because I may buy your game and play it ☺

Contact Information:

E-Mail: m.hartl@fraktalia.org

Homepage: http://fraktalia.org/