Brute Force - Sand Shader Documentation

1. Brute Force - Sand Shader

Brute Force - Sand Shader Unity Asset Store

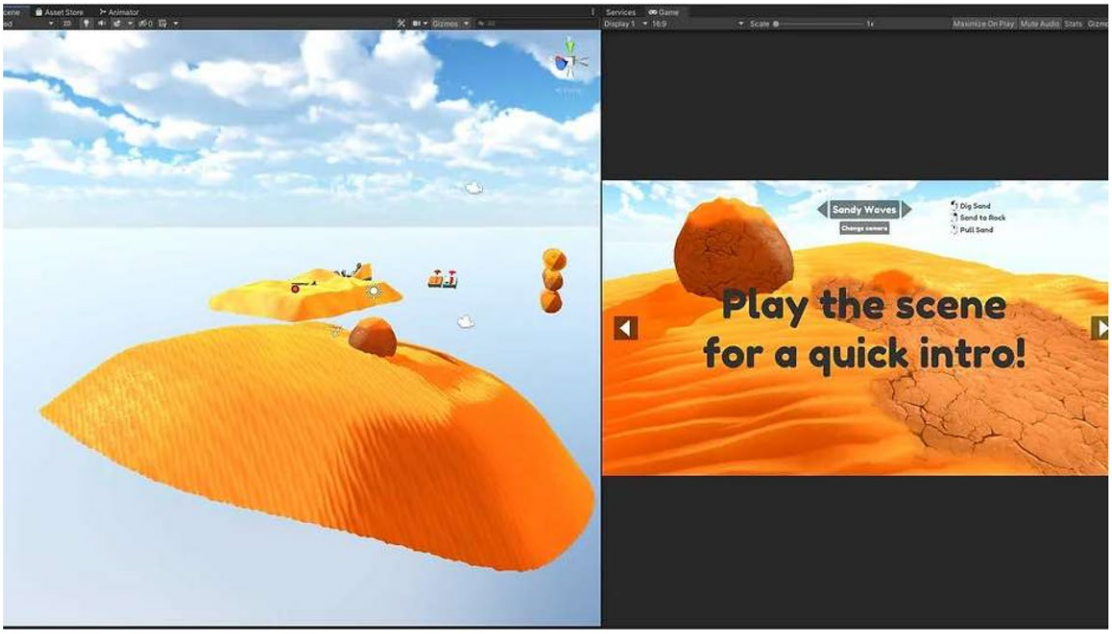
This is the **Brute Force - Sand** visual documentation.
Consider leaving a review on the asset store to support me c:
and if you have questions email me at: **bruteforcegamesstudio@gmail.com**

3. Compatibility and Features

The **Brute Force Sand Shader** let's you create an **interactive sand** combined with **rock textures** for **PC/Mac/Linux/Nintendo Switch** and **mobile** devices.
Compatibility with **Unity Standard** and **URP**

Features:
-**Mutlicompatibility**: works on PC/Mac/Linux/Nintendo Switch/WebGl and mobile devices
-**Simplicity**: Drag and drop materials
-**Tessellation**: Custom tessellation based on camera distance
-**Dynamic Custom Shadows**: Sand featuring custom colored shadows and shadow casting
-**Interactivity**: With a simple particle system you can create interactive effects like a trail on mouse/player position
-**Customization**: Fully customizable look, render the sand with a single material
-**Additive Sand**: An experimental feature letting you add sand to any mesh with one simple component
-**Lights Support**: 4x point lights and spot lights support (with shadow casting)
-**Terrain**: Use texture Splat Map to draw sand terrains, the asset features a compatibility tool to work with other terrain assets.
-**VR Support**: Works with Multi and Single Pass Stereo rendering

4. Getting started



First thing you should do is load up the sand demo scene inside the asset package
"Assets\BruteForce\Scenes\StandardSand"
You will see that some files have a Standard and URP version, choose accordingly.

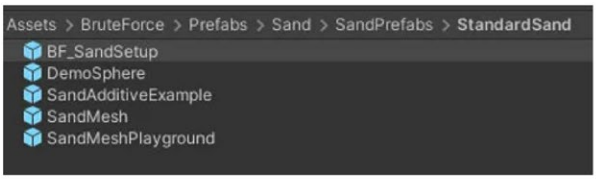
Once the **"01_Sand"** or **"01_SandURP"** is loaded you will be prompted to play the scene for a quick introduction. This will help you understand the asset better and showcase all its basic features in a complete environment.
In the first showcase you can use **Left Click** to dig the sand, **Right Click** to change sand into rock, **Middle Click** to pull sand upward.

You can click the top arrows on the screen to switch sub-showcases and the left and right arrows to switch showcases.

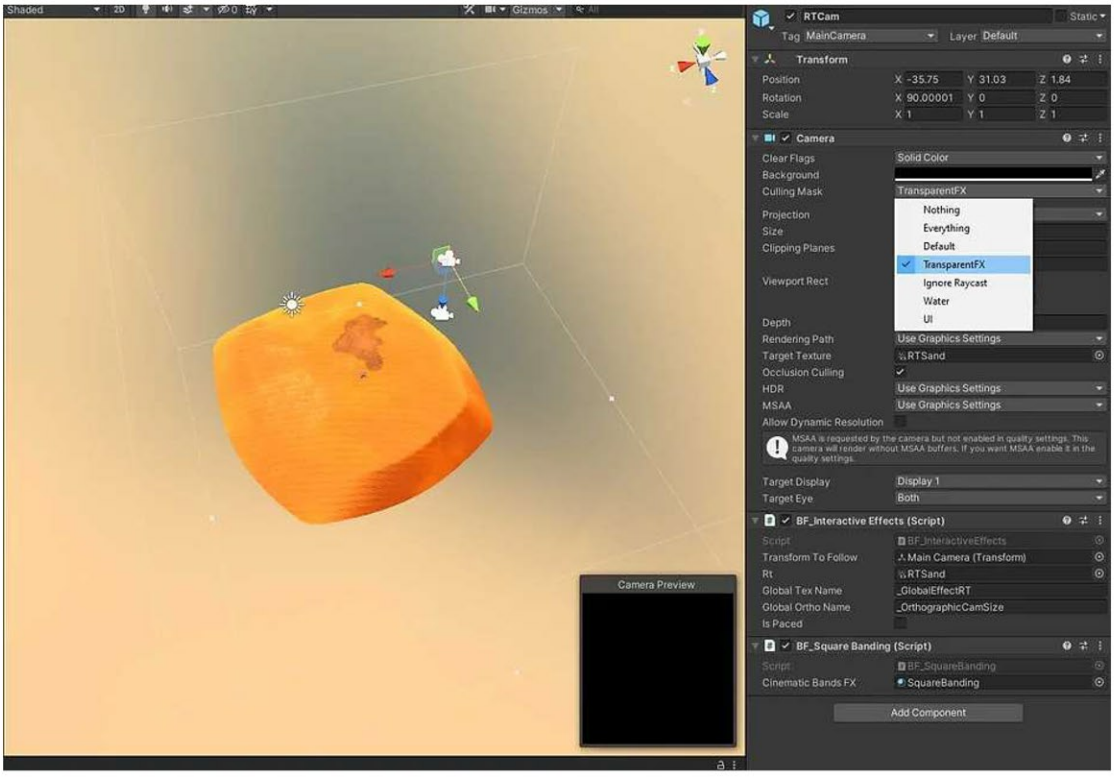
Additionally you can use WASD to move the mini rover when showcased, it works great as a player template leaving an interactive trail on the sand.

Importing it in your project and Components

To append the sand assets in your projects you can either copy and import one of the working **showcase** as a starting template or you can use the **setup prefab**. Keep in mind that if you need the interactive feature you have to append the **CameraEffects** in your scene. I will go into details on what components you need in the prefab for the sand to work properly:



Camera Effect

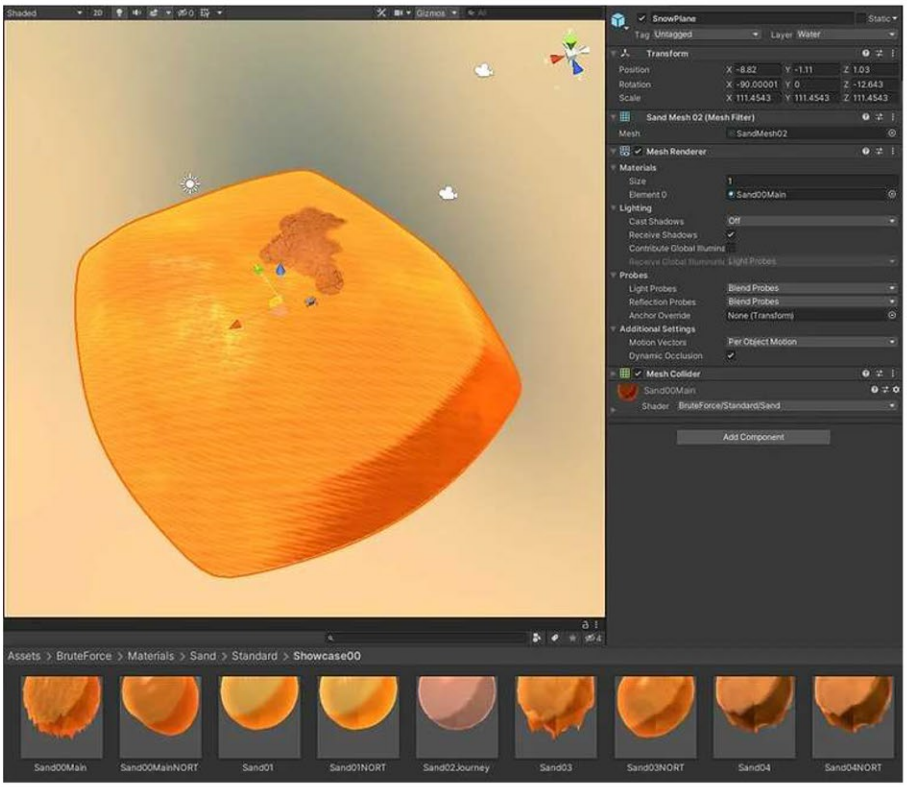


The first main component you'll need if you want **interactivity** on your sand are the "**Camera Effects**" like a regular camera it will be used for rendering, but instead this one will render to a target texture namely: "**RTSand**" and "**RTSandAdditional**" and will have only ONE culling mask set to "**TransparentFX**", we only need to render the effects.
It needs to have a target texture set to a render texture and it also needs the script "**BF_InteractiveEffects.cs**" for the first one and "**BF_InteractiveEffectsAdditional**" for the second one (which is optional, more details below). This script will

make sure your cameras will render to the render textures.
If you have a moving camera you'll need to set a transform to follow (preferably the camera). RTCamAdd MUST have your main camera as transform value.
If you don't know what the global names are do not change them.
If you do not need the interactive part of the shader you can disable the CameraEffects and the "**Use RT**" in the material pragmas for a small performance boost.

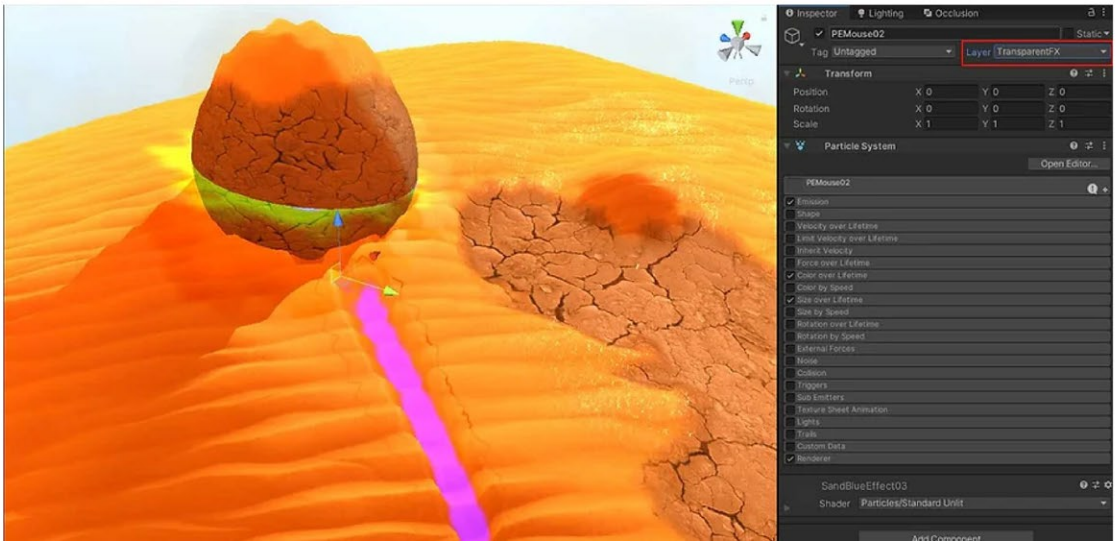
You can increase the size of the camera projection to increase the radius of the rendered effect at a precision cost (more details below), do not rotate the RTcamera.

Sand Material



You can drag and drop any sand materials to a mesh or any assets you want to use as ground, I will explain in more depth how the Sand Shader works in the next page so you can be able to customize or create a sand material for your projects.

Particle System Effects



To control the interactive effect of the sand you can use a particle system set on the **TransparentFX layer**, the important

thing to note is that only the rendered color matters:

Red color will turn snow into rock, **Blue** color will dig the sand, **Green** color will pull up the sand.

You can use anything that can be rendered to achieve the desired effect: meshes, sprites, particle systems etc...

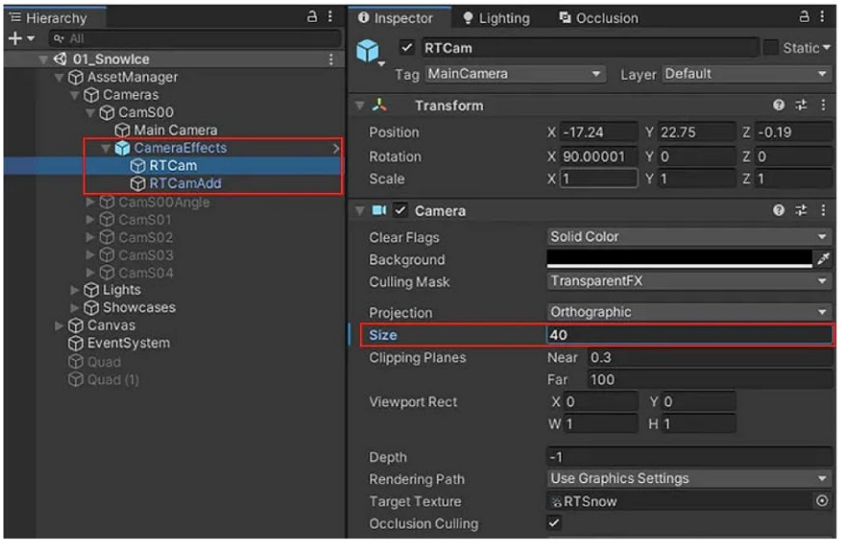
This means you can place a **colored quad** in the **TransparentFX** layer to have a **permanent sand effect** in your scene.

Alternatively you can edit the "Duration" and "Start Lifetime" parameters of the particle system to either shorten or extend the effect duration.

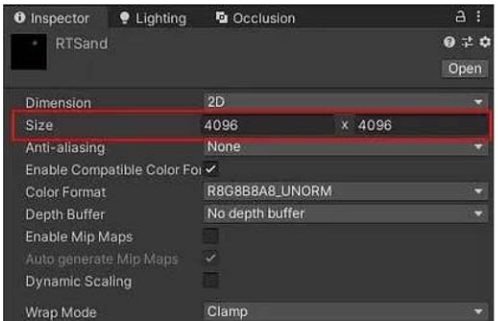
Increasing the Render Texture size

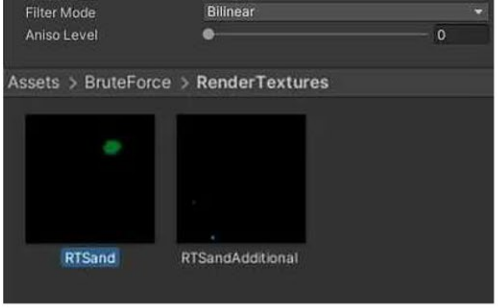
Depending on your project, you can increase the size of the Render Texture field so you would be able to see sand effects from much further away.

To do so you'll need to select your **RTCam** in your scene and increase the "Size" Camera variable, keep in mind that increasing this value will decrease the effect resolution.



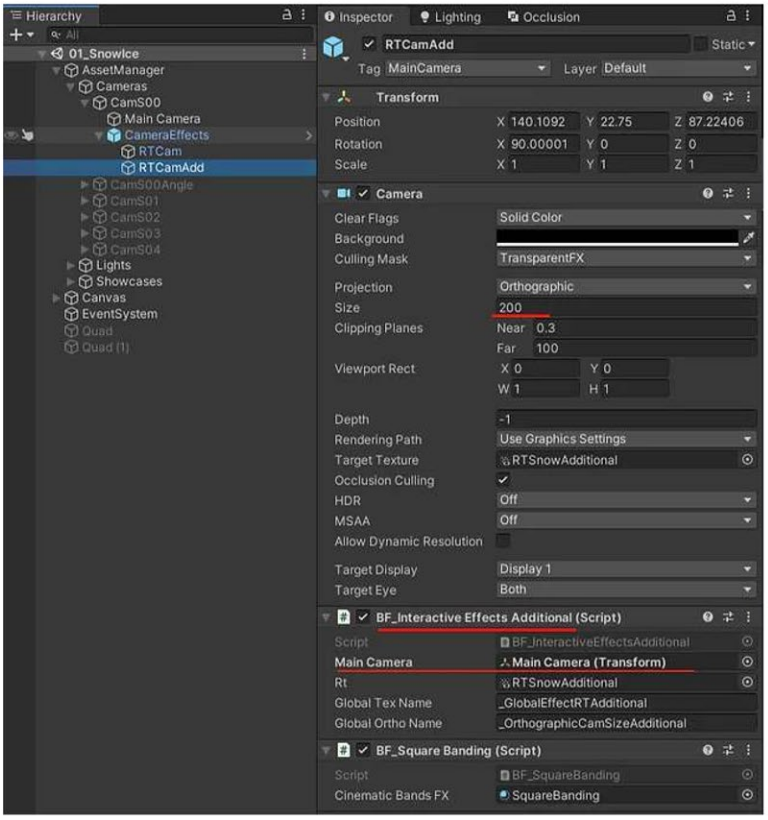
To increase the effect resolution you can change the "**Size**" of the **RenderTexture** to 2048x2048 or 4096x4096 by selecting the "**RTSand**" in your project. Changing this value will greatly increase memory consumption.





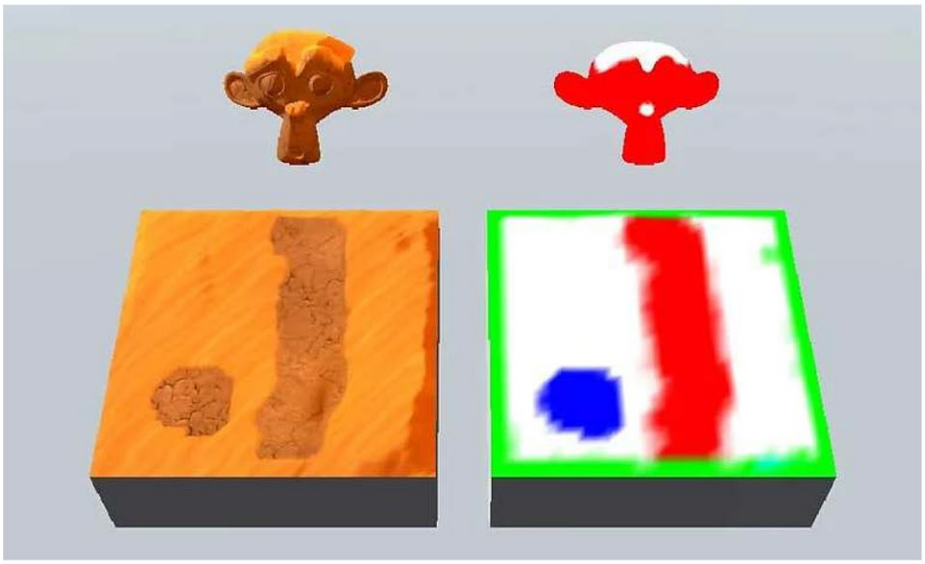
Using the RTCamAdd to expand the Sand effect area

In addition to the RTCam I created an experimental additive RenderTexture effect which is much larger. This camera is optional but its use is highly recommended if you intend to use the effects on a large area.



When you assign the main camera to the "Main Camera" field in the **BF_InteractiveEffectsAdditional** script; the RTcamAdd will automatically change its position based on the main camera FOV.

5. Paint Sand

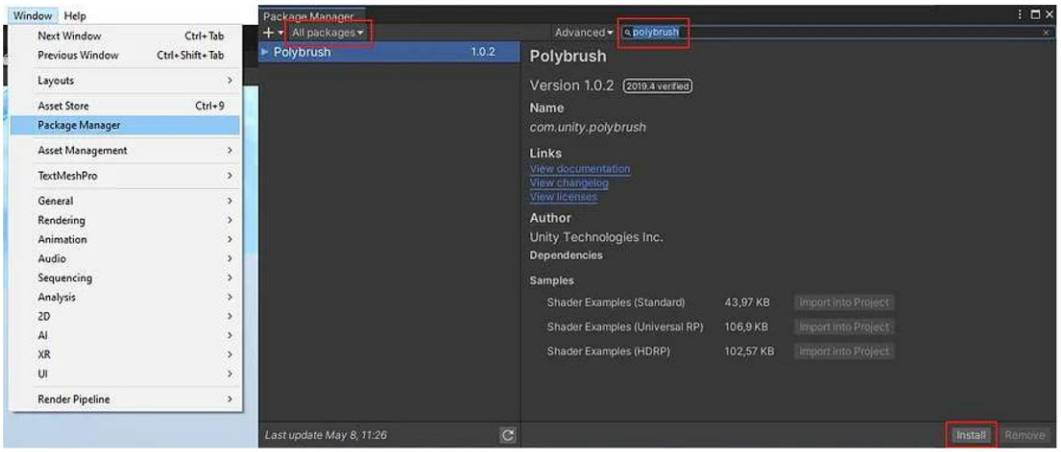


Using the Brute Force - Sand shader you can paint the vertex color of your assets to dynamically draw sand or rock.
The default value and state of the shader is sand.
With that in mind let's see in details what each vertex color channel do:

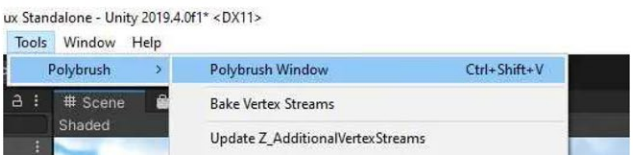
-**White**: Renders Sand
-**Red**: Renders Rock
-**Green**: Renders Sand without displacement
-**Blue**: Renders Rock without interactive effects

How to paint Vertex Color on your mesh

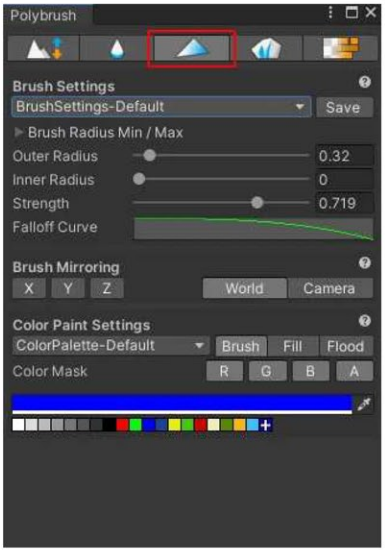
I strongly recommend you download the Polybrush Asset from the Package manager (see documentation). It's an official Unity asset that lets you draw colors on vertex. You can also use your favorite 3D software to draw vertex color and it will work fine.
For the rest of this page I will assume you are using the Polybrush asset.



Now that you've downloaded and installed **Polybrush**:

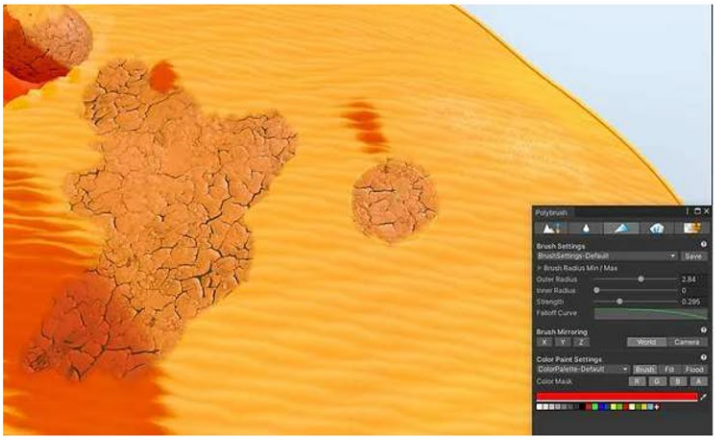


Open the Polybrush Window by going to: **Tools** > **Polybrush** > **Polybrush Window**
You will have this window opened:

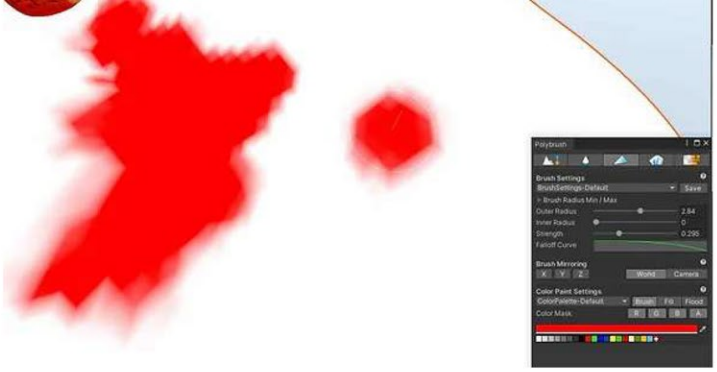


The tab that interests us is the "**Paint vertex colors on meshes**" click on that.
You can see there is a "Outer Radius" value that controls the radius of the brush size and a Color Paint Settings at the bottom.
You can choose which color you want to paint by clicking the big color field below color mask.
When choosing between White, Red, Green or Blue colors make sure it is the absolute color; meaning white is fully white (1,1,1) and red is fully red (1,0,0). Again, tead the polybrush documentation for more details.
Select red color:

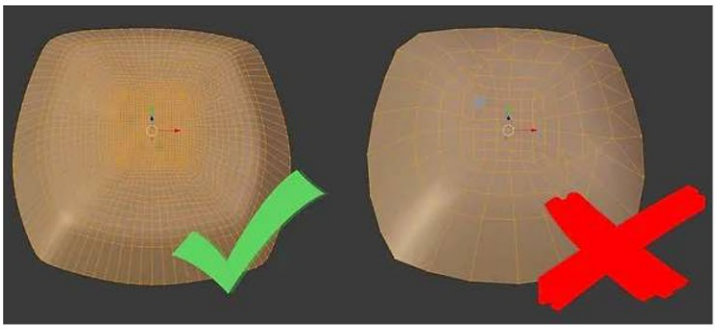
Now click on the mesh you want to paint on with the sand material already placed, you can start painting rocks:



This is what you are actually doing by painting:

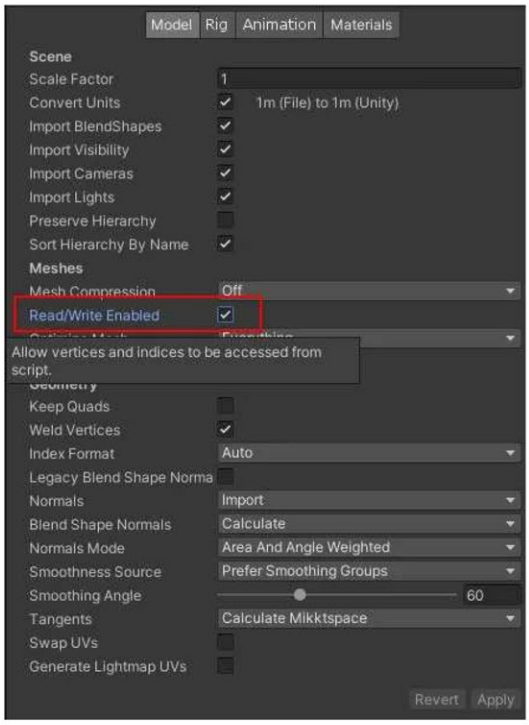


Keep in mind you need a certain **number of vertices** to have a higher effect definition, here's a good guideline:

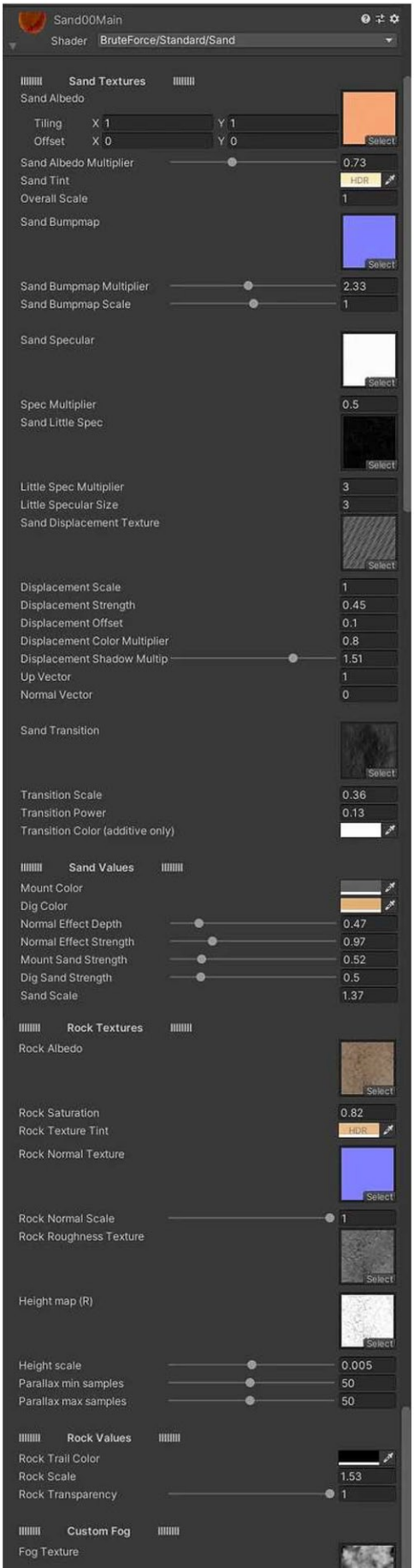


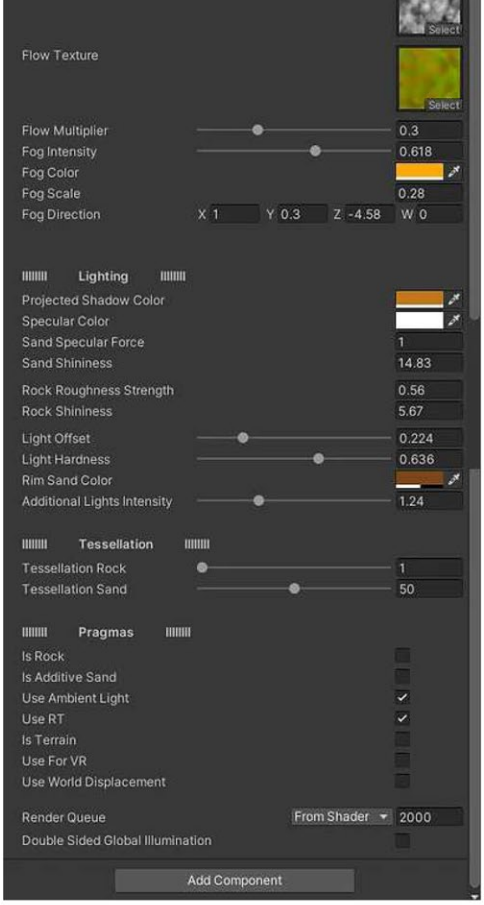
Finally when you are happy with your mesh make sure the "Apply As" is set to **Overwrite Mesh**

When importing a custom mesh inside Unity **you are required to enable Read/write** in the import settings of said mesh.



6. Sand Properties





The Sand asset is controlled entirely through the material property block, I suggest toying with the values to familiarize yourself with them. The material being very long I categorized the fields into:
Sand Textures || Sand Values || Rock Textures || Rock Values || Custom Fog || Lighting || Tessellation || Pragmas

I will describe the important parameters in order:

|| Sand Textures ||
- **Sand Albedo**: A colored texture to control the sand color
- **Sand Albedo Multiplier**: The saturation of the sand texture
- **Overall Scale**: The scale parameter controlling every texture coordinates except the sand displacement
- **Sand Little Spec**: The B&W texture controlling the specular intensity where the light reflects on the sand surface
- **Sand Displacement Texture**: The texture controlling the displacement of the sand vertex
- **Displacement Offset**: The value differentiating the height between sand and rock, 0 = no differences
- **Displacement Color Multiplier**: The parameter multiplying the value (color) difference based on displacement height
- **Displacement Shadow Multiplier**: The parameter multiplying the shadow value based on displacement height
- **Up Vector**: The higher this value is the more the displacement will move towards the world upward vector
- **Normal Vector**: The higher this value is the more the displacement will move towards the vertex normal
- **Sand Transition**: The texture controlling the transition shape between sand and rock

|| Sand Values ||
- **Mount Color**: The color tint of the pulled up sand based on interactive effect (green)
- **Dig Color**: The color tint of the dug up sand based on interactive effect (blue)
- **Normal Effect Depth**: The multiplier offset of the interactive effect, does not scale with the RenderTexture dimension if you do not understand what it does leave it between 0.10-0.20
- **Normal Effect Strength**: The value multiplying the overall interactive effect
- **Mount/Dig Sand Strength**: The vertex displacement multiplier of the respective effect (Vertex position only)
- **Sand Scale:** The scale parameter controlling the sand texture coordinates

|| Rock Textures ||
- **Rock Albedo**: A colored texture to control the rock color
- **Height Map**: The texture controlling the parallax occlusion effect, white = no height, black = deep height
- **Parallax min/max samples**: The sample definition of the height effect, lower it for better performances

|| Rock Values ||
- **Rock Trail Color:** The tint of the interactive effect on the rock (color is reversed; white = very dark tint trail)
- **Rock Scale:** The scale parameter controlling the rock texture coordinates
- **Rock Transparency:** Parameter used by the additive component and the terrain script, sets the rock alpha, in most cases you do not want to touch this

|| Lighting ||
- **Projected Shadow Color:** The color of the sand & rock shadows
- **Rim Color:** The tint color of the sand rim light (alpha controls the scale)
- **Light Offset/Hardness:** Range values to control how the light gets rendered on the material

- **Additional Lights Intensity:** The additional lights (points, spots) intensity multiplier, only works with additional lights

|| Tessellation ||

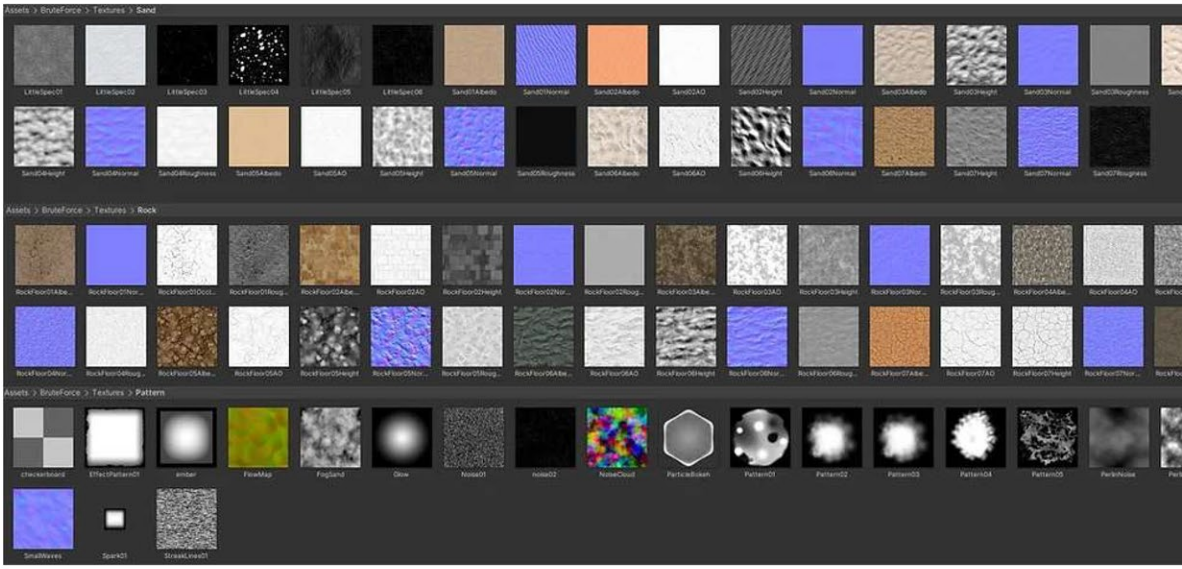
- **Tessellation Rock:** The tessellation control value of the ice, 1 = No Tessellation | 100 = maximum Tessellation, in the majority of uses you SHOULD leave it to 1
- **Tessellation Sand:** The tessellation control value of the sand, a good optimized value should be between 20 and 50

|| Pragmas ||

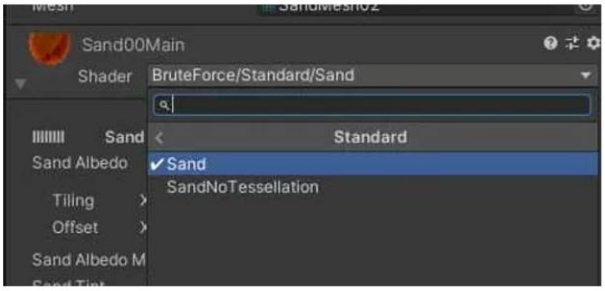
- **Is Rock:** If enabled it will turn the whole mesh into rock independently from vertex color
- **Is Additive Sand:** If enabled it will activate the additive sand feature culling the rock, do not enable this feature by itself instead you should use the Additive Sand component
- **Use Ambient Light:** Enables the use of ambient scene lighting
- **Use RT:** If enabled it will activate the interactive feature, disable it if you don't intend to use the interactive part
- **Is Terrain:** If enabled it will activate the Terrain feature, do not enable this feature by itself instead you should use the Sand Terrain component
- **Use For VR:** Enables the compatibility with single pass rendering for VR
- **Use World Displacement:** The If enabled it will project the sand displacement texture in world coordinates independent from mesh uv, turn it off if you want to use your mesh uvs for sand displacement

You can switch textures around or try different materials from the large collection in the package:

You can create your own textures for any texture slots



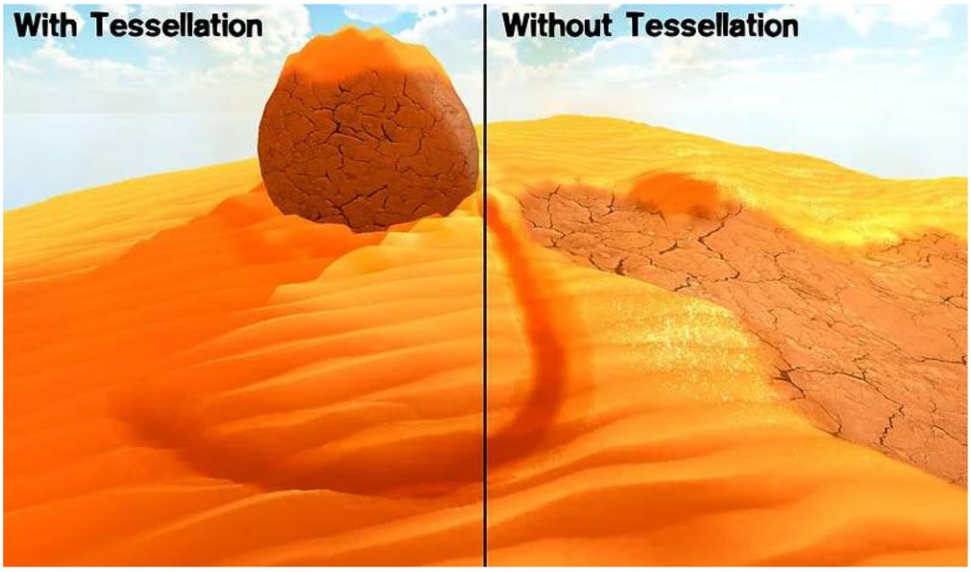
If you want to create your own materials here is the list of the Sand shaders:



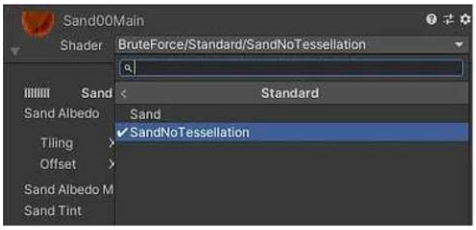
Sand: The standard Sand shader

SandNoTessellation: The same as the above but without the vertex tessellation feature, use this variant to build for WebGl or low-end mobile; see more about WebGl and Mobile Build

7. WebGl and Mobile Build



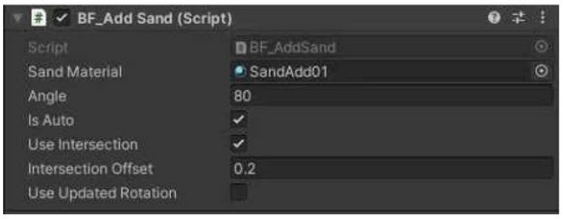
If you want to use the Sand shader for your WebGl or low-end mobile projects the asset features a tessellation-less variant of the shader that is compatible with low render target APIs. Simply switch any sand materials to the variant:



As you can see in the image above you will not lose much fidelity and you will gain compatibility and performance.

8. Additive Sand Feature

The additive Sand feature is an experimental component that you can attach to any meshes to add procedural sand to them:



Sand Mat: The material to attach to the procedural mesh

Angle: The value controlling the procedural calculation of snow area, if **IsAuto** is disabled you can ignore this value. The higher the value the more faces that look in another direction than upward will have sand

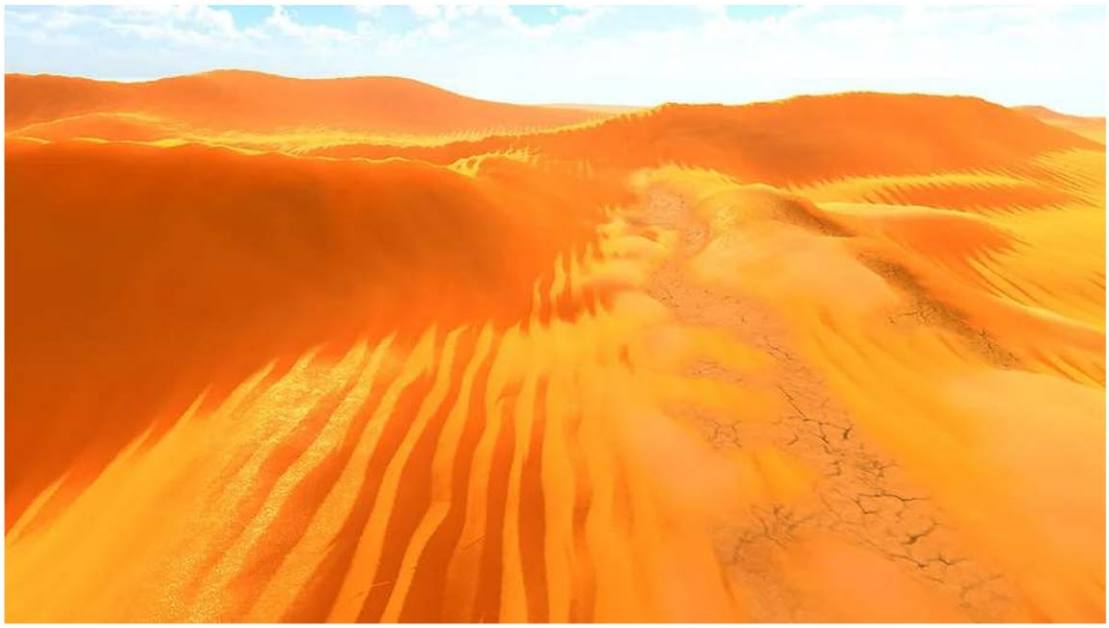
IsAuto: Is the area covered by sand calculated procedurally based on original face normal, if this is set to false the calculation will take the original mesh vertex color to render the sand area so make sure to have painted you mesh before disabling this option

Use Intersection: Is the intersection feature being used, if you enable this pragma the additive mesh will raycast the nearest surface and automatically render a sand gradient on the mesh

Intersection Offset: The offset value of the intersection gradient

Use Updated Rotation: Does the additive mesh updates on rotation

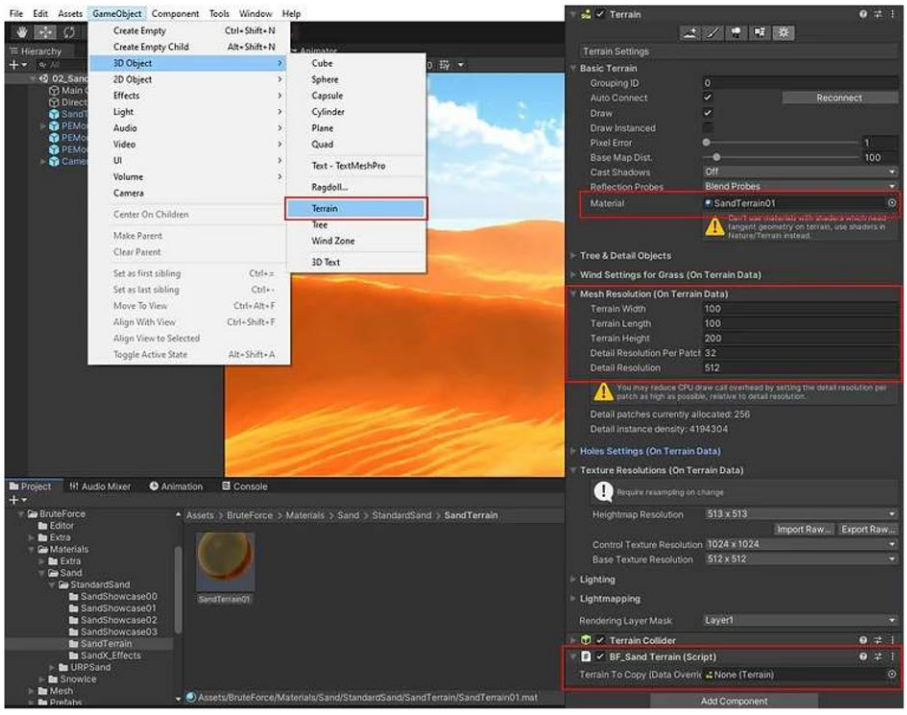
9. Terrain Feature



There is a template scene for terrain environment in the asset package (both built-in and URP) located here:

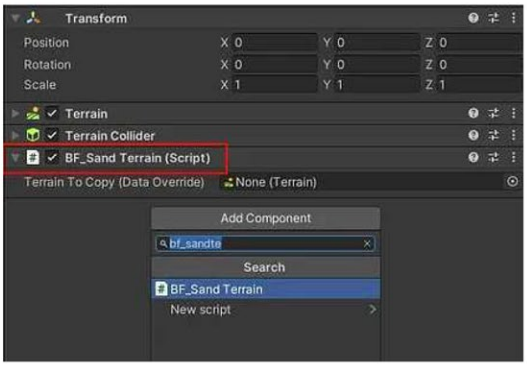
If you want to create your own terrain and implement the sand shader here's the important steps to follow:

1. Setup the terrain



Create a new terrain with a 100x100 dimension (works best out of the box rather than the HUGE 1000x1000 default terrain)

Then add the "SandTerrain01" material or its URP equivalent or your own Sand material

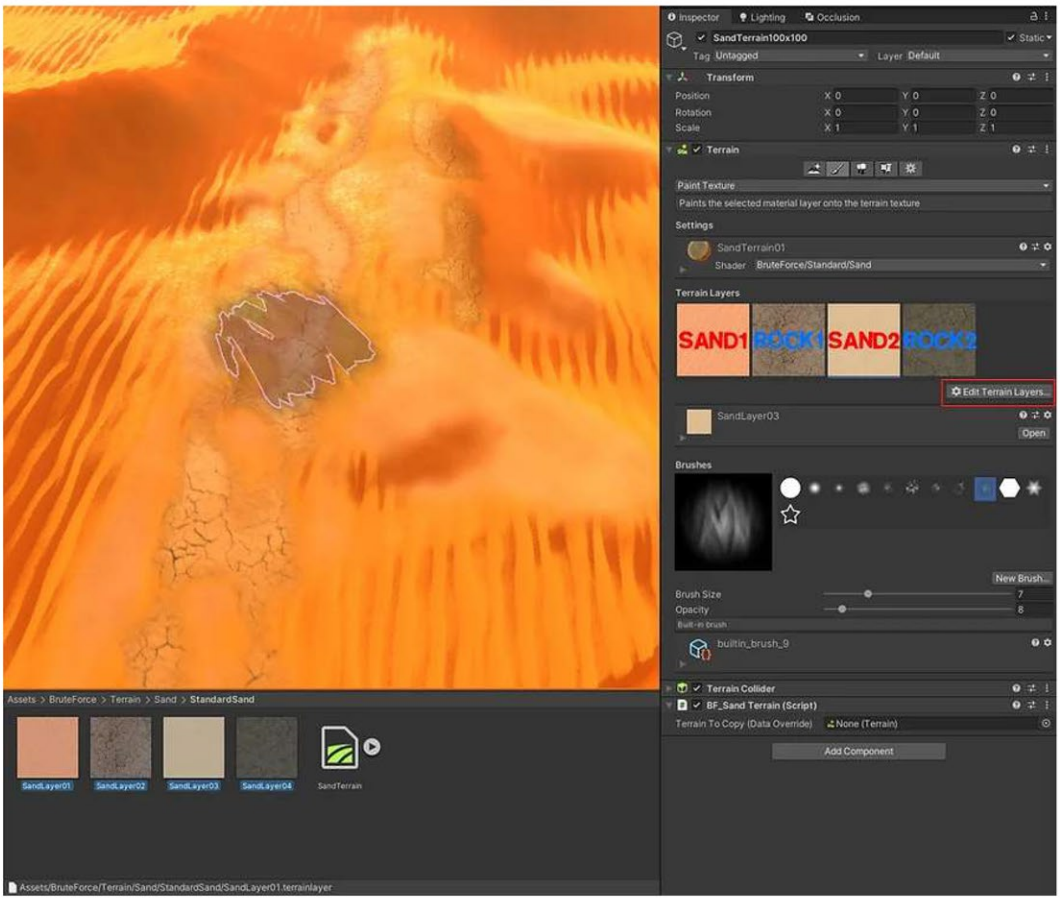


IMPORTANT: You need to add the "**BF_SandTerrain**" script to your terrain for it to work properly and if you're only using the terrain feature of the sand shader without regards for other terrain assets compatibility you are not required to add a "terrainToCopy" parameter nor should you press the "Sync terrain Data" button

VERY IMPORTANT: Be extra careful with these fields, if you assign or press the sync terrain button on a terrain you do not want to change its previous height map it will override and change it, you can press the revert button to set the values back to how they were.

2. Edit terrain layers

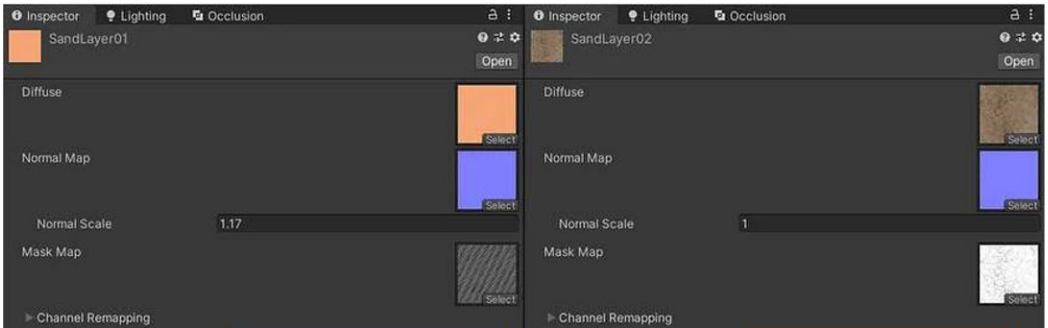
After you've set the right shader/material to your terrain get over the paint terrain tool in the terrain inspector:

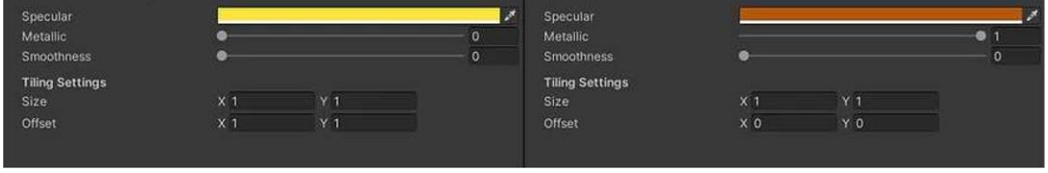


To draw snow on the terrain use the **SandLayer01**, the position in the terrain layers arrangement is very important, the first one SHOULD ALWAYS be the sand layer.

To draw rock on the terrain use the other **SandLayer** (02,03), if you don't want/need other sand/rock layers you can delete them via the "Edit Terrain Layers" button (Currently limited to 4 layers).

If you want to edit the layers to have a new sand/rock layer you should head over to the corresponding "**NewLayerXX**" in the asset project:





The first layer which should ALWAYS be a sand layer;

If you want to have a Sand Layer:
Diffuse is the Albedo Texture,
Normal Map is the bump map,
Normal Scale is the normal multiplier,
Mask Map is the displacement map,
Specular is the overall color of the sand,
Metalic should be set to **0**,
Size.XY is the Tiling size

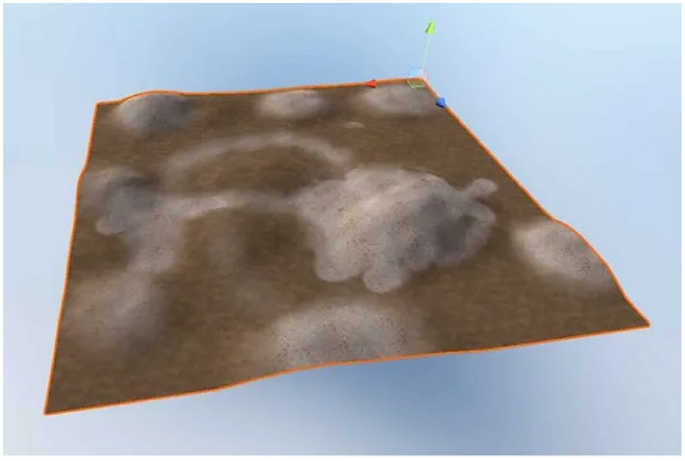
If you want to have a Rock Layer:
Diffuse is the Albedo Texture,
Normal Map is the bump map,
Normal Scale is the normal multiplier,
Mask Map is the Height Map,
Specular is the overall color of the rock,
Metalic should be set to **1**,
Size.XY is the Tiling size

The only difference between a Sand Layer and an Rock Layer are Mask Map and Metallic slider.

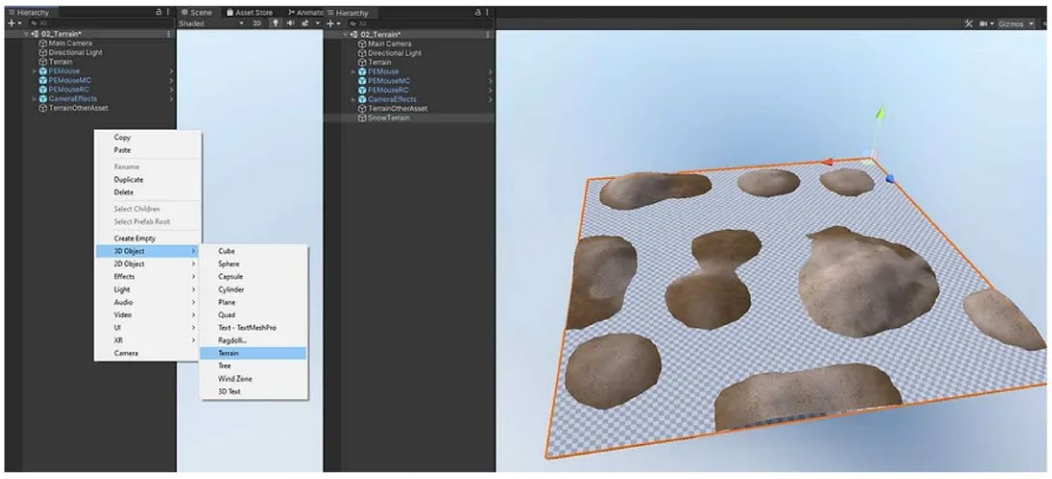
STOP there if you don't use any other terrain assets and are not looking for cross-compatibility!

3. Terrain Asset compatibility (optional)

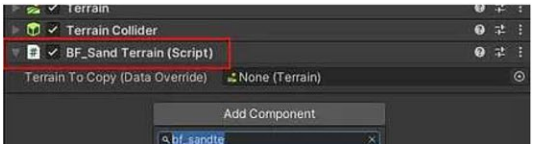
I implemented a solution for users to use my sand shader with any other terrain assets out there.
Here is what you need to do in order to get it working:



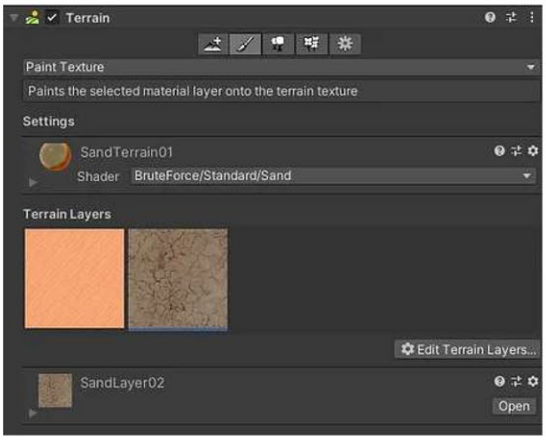
Let's assume you have a working terrain working with another terrain asset (it must be using the built-in terrain as a base)



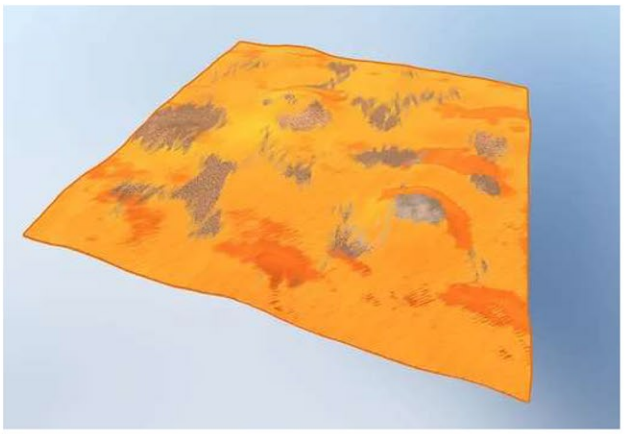
Setup the sand terrain just like described in the steps before, add the sand terrain material and the **BF_SandTerrain**.cs script.



In the **BF_SandTerrain** component; drag and drop the other asset terrain in the "TerrainToCopy" field.
Then press the Sync Terrain Data button to sync the dimensions and height texture.
(I strongly recommend you to lower the tessellation count on the sand terrain material, you don't need as much precision for sand on a terrain)



The first layer should be a sand layer with metallic set to 0 just like the regular setup. And the second layer has to be a rock layer with metallic set to 1, use this layer to remove sand of the first layer.



Now you should be able to draw sand with the sand layers and cull the sand to show the other terrain asset with the second layer of rock! You can add sand layers to this process for more variety.
Please keep in mind that these steps are only necessary when you want to use my sand shader terrain feature with other terrain assets, you do not need to do this if you only use my shaders.

If you're having trouble setting up a terrain with the sand shader here's a step-by-step video to complement the informations above:

10. Performance Tips

Tessellation Factor

Probably the biggest performance factor of the sand shader is the tessellation factor.

You should keep the **Tessellation Rock to 1** (which is equal to no tessellation)

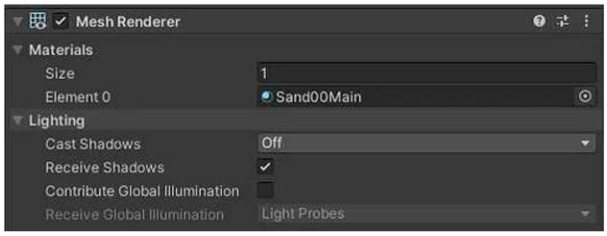
For PC/Linux I recommend setting **Tessellation Sand between 20 and 50** and for mobile/VR/AR you should stay **below 20**.

The performance hit is relative to the view distance between the sand and the camera.

Material Pragmas

Check or uncheck the features you don't need in the material, if you don't need Interactive effect you should disable "Use RT".

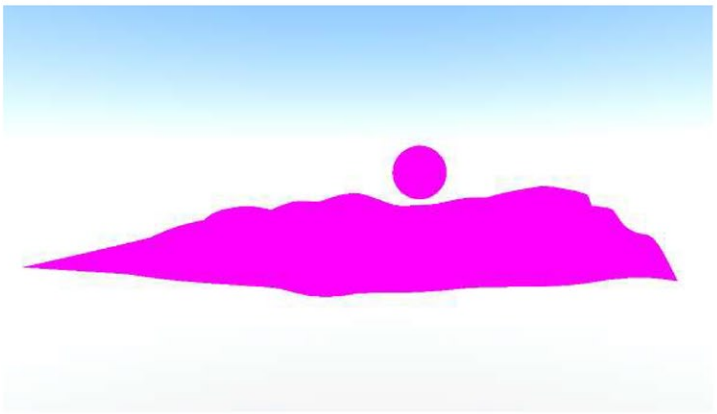
Disable Shadow Casting



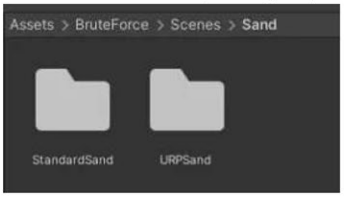
You might not need the shadow casting feature of the mesh renderer in which case this could save you a good performance chunk. This is especially true for terrains which have a Double shadow casting enabled by default!

11. FAQ & Troubleshooting

I opened the package scene and all I see is pink!



You opened the wrong scene, there is 2 variations of the Sand Shader: one for Unity Standard and one for URP.
Use the folders corresponding to your render pipeline:

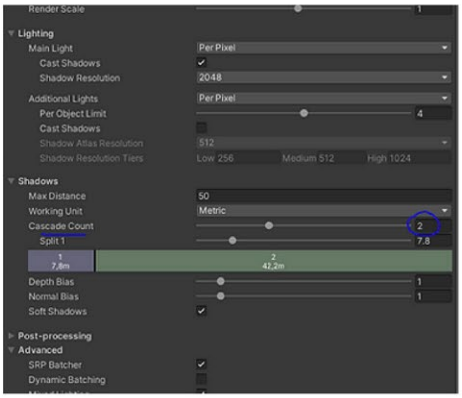


I can't find the URP folders and when I import the package it says only v1.0

Contact me at bruteforcegamesstudio@gmail.com with your unity asset store order and I will give it directly to you, chances are the unity package manager is forcing the wrong one on your unity version.
You can solve this issue without my help by installing any unity above 2019.4 and importing the package, then you can export the unity package in your projects.

I have shadow artefacts on Mobile URP

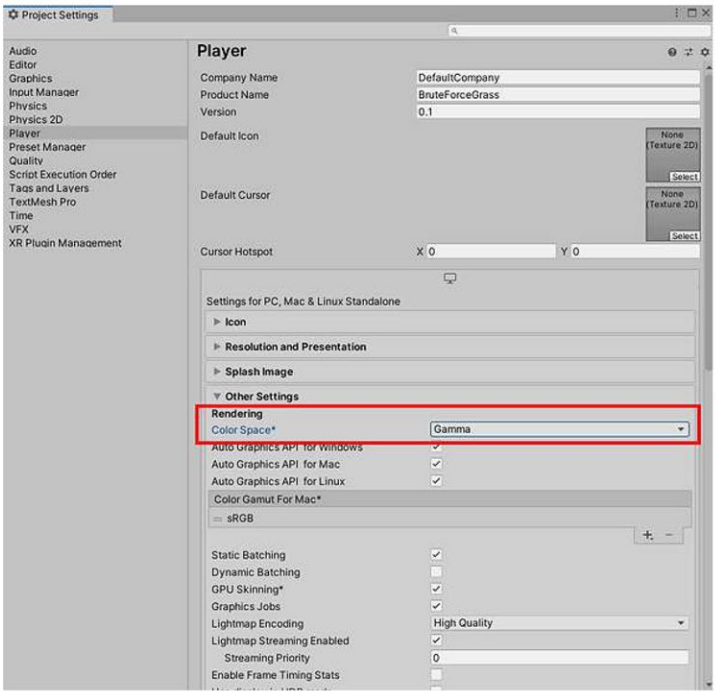
The problem comes from the URP pipeline itself, to get around this issue you need to set your shadow cascade count to 2 or more:



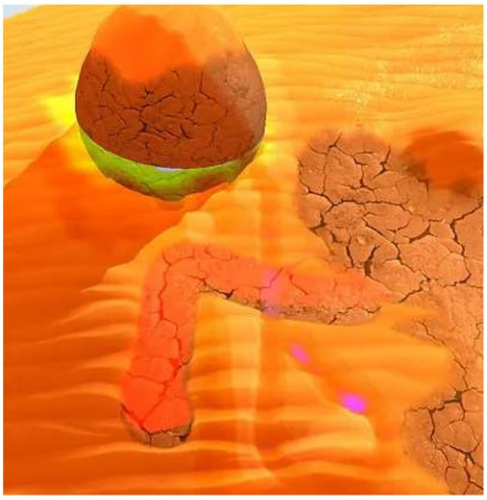
The sand looks so pale/bright in my project

Color differences may occur when you're not using the default unity color space depending on your Unity version and Render Pipeline, (Gamme for Built-in and Linear for URP)

Either change the rendering color space or simply tune the materials to look the way you want

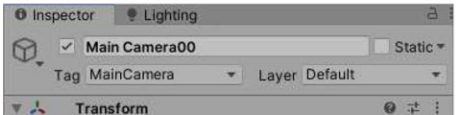


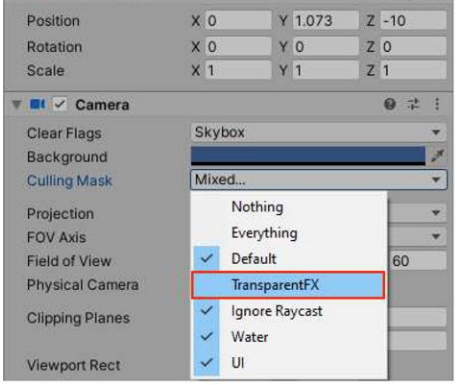
The Particle Effect is visible in the editor or at runtime



If you want to not see the particle effect rendering inside the editor, you have to disable the same layer it's in by using the layers visibility option, here the layer used is **TransparentFX**:

And if you need to disable it's visibility inside the game window or on build you need to do the same for the camera culling masks of your main camera:




Position
X 0
Y 1.073
Z -10
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1
Camera
Clear Flags
Skybox
Background
Culling Mask
Mixed...
Projection
FOV Axis
Field of View
60
Physical Camera
Clipping Planes
Viewport Rect
Nothing
Everything
Default
TransparentFX
Ignore Raycast
Water
UI