Insta Polish Documentation

Note This asset is not for absolute beginners , using this asset requires at least some degree of unity experience (intermediate experience level is recommended). The goal of this asset is to give instant Polishing effects that will increase the polishness and realism of the game (attention to detail).

This asset tries to strive for drag and drop structure so it won't interfere with your existing systems. Even shaders that we believe core is made into module sub graphs so it can be easily integrated into your own shaders . Also, other scripts are also made in such a way that they work as extensions , not as a core. Also this asset target for indie games or start point for AAA games. Disclaimer ! do not use this pack effect as core gameplay mechanic in your game without any modifications (we believe these features are not robust and good enough for a core feature of a game).

This asset is perfect for game jams and indie games with short deadline.

Important Note
This pack is not about reinventing the wheel — many of these effects have been demonstrated in tutorials or could be recreated by experienced developers.
However, the **Insta Polish** pack focuses on delivering:

- **Time Savings** – All effects are pre-built, refined, and ready to use, saving hours (or days) of setup.

- **Enhanced Features** – Each effect includes additional controls, utility scripts, and flexibility beyond typical tutorial implementations.

- **One Unified Package** – Brings a variety of scattered effects into one consistent, performance-friendly bundle.

- **Up-to-Date Compatibility** – Fully tested with Unity 6 and modern render pipelines, avoiding common issues found in older tutorials.

- **Polished Results** – Effects are optimized, visually tuned, and designed for ease of integration into existing projects.

If you need proven, production-ready polish effects without the hassle of piecing them together yourself, this pack is designed for you.

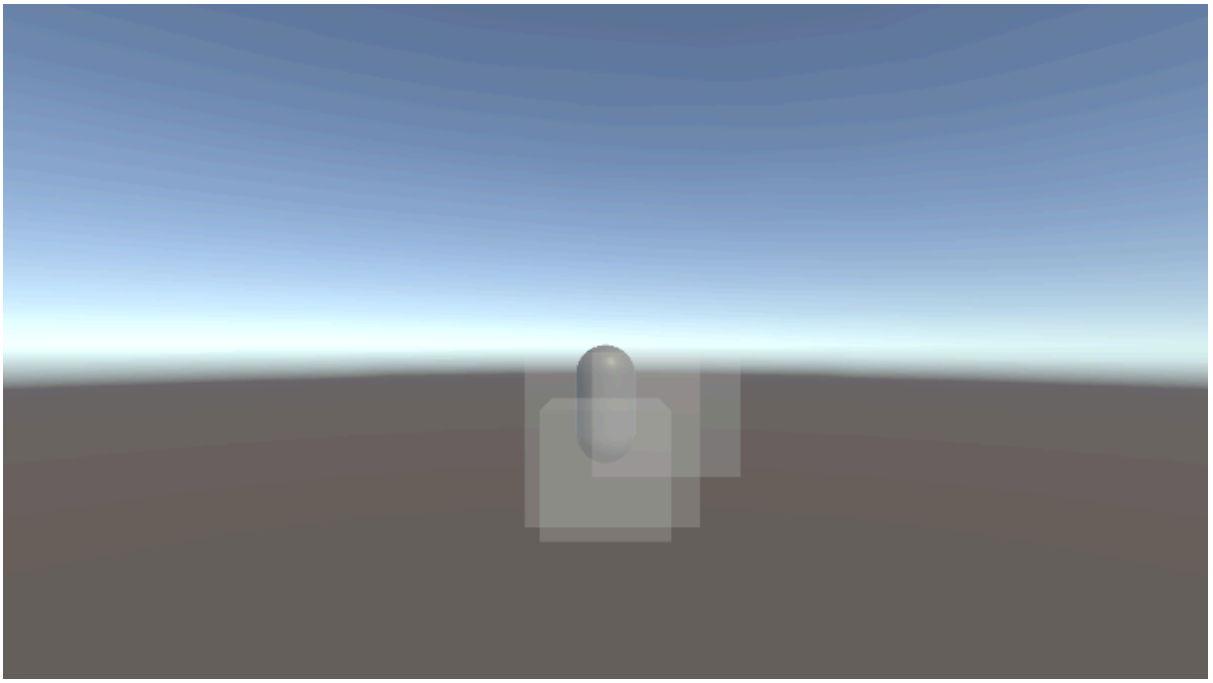
Camera Block Effect

This effect is simply fading the alpha of objects that stand in the way of camera and player , thus the gamer would have always player in sight.

Important : this effect use material property block so you don't have to worry about material instancing or increasing draw calls

To use the effect simply add camera obstruction fade script to your main camera. Then adjust the properties according to your likeness and press play and it should work. See the example camera fade scene. Note surface color properties should be a color and script only changes its alpha channel.
Also note every potential shader keyword that needs to be faded in the script.

You can hover above the script properties and the tooltip will say what it will do.



Following image shows player capsule is blocked by 3 cubes and camera obstruction fade is dynamically changing the alpha so player is always visible

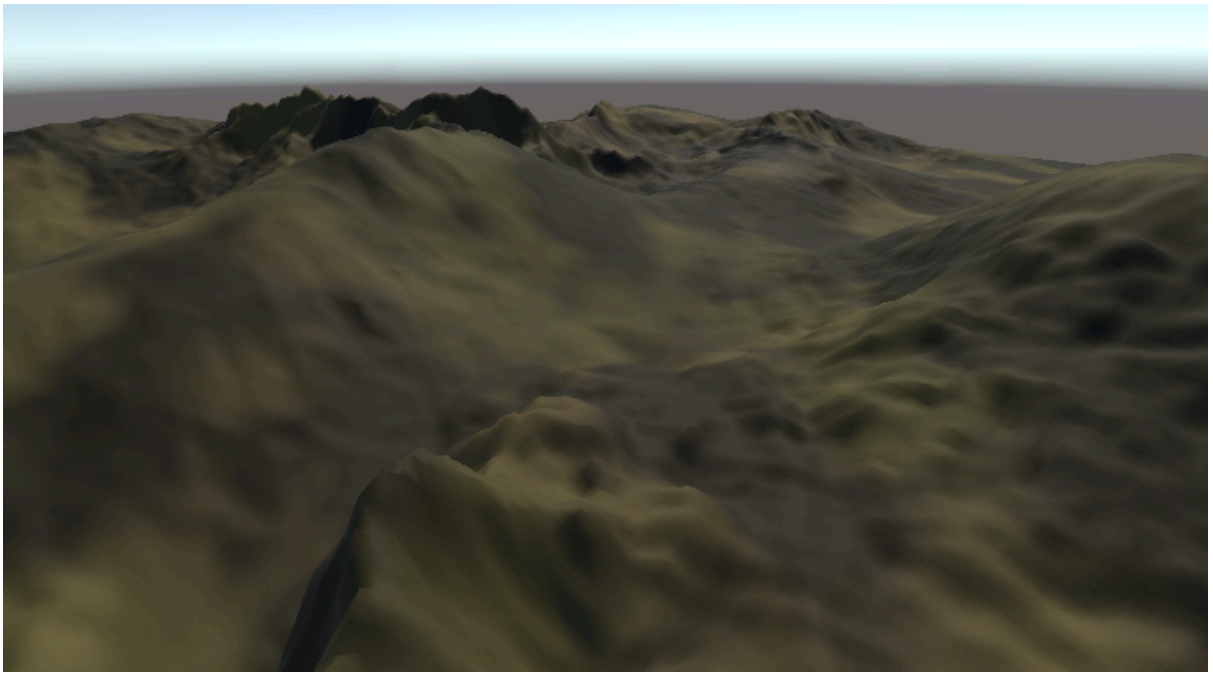
Cloud shadows effect

This effect will create fake cloud shadows that will move in the terrain breaking the repetition and increasing the realism , also with this asset you can use just skybox with cloud without actually creating clouds that will do shadow cast (creating actual clouds will make things realistic but it is also performance expensive).

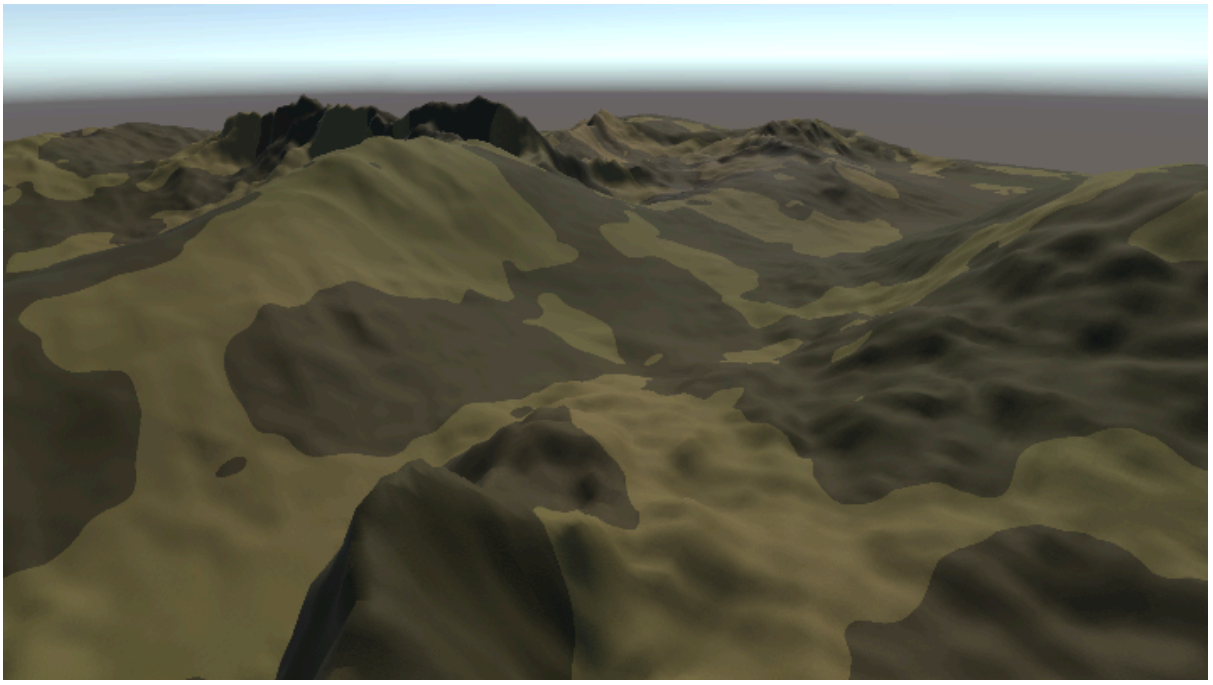
This cloud shadows is using a decal projector and for every new scene you need to add new decal projector that will cover the entire map.

Important - to change the cloud shadows property go to the decal projector select the material and change its properties (play with values until you get satisfying results).

The following image shows a terrain with subtle, realistic cloud shadows. The shadows are intentionally light to blend seamlessly with the environment.



The following image shows the same cloud shadow effect but for stylized toon games where gradients are not present. You can customize these properties from the cloud material.



Dissolve shader

A dissolve shader made with dissolve sub graph , this demo scene has pre built script with editor buttons (for debugging) and public methods that can be called on demand via external scripts.
There is 2 animatable parameters in the shader which are dissolve level which controls noise dissolve and vertical movement which control is dissolve should happen vertically . Both are animated by the script

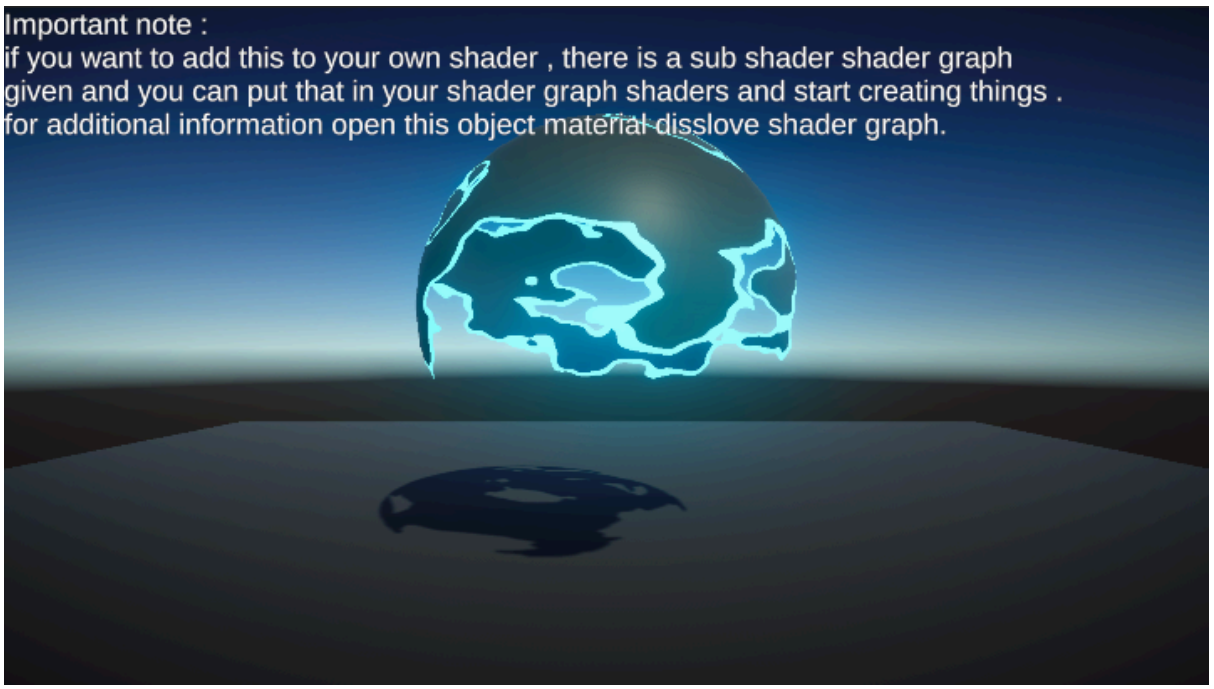
Public method to call

Spawn - use to spawn the (or un dissolve) object
DeSpawn - use to Despawn the (or dissolve) objects

Note

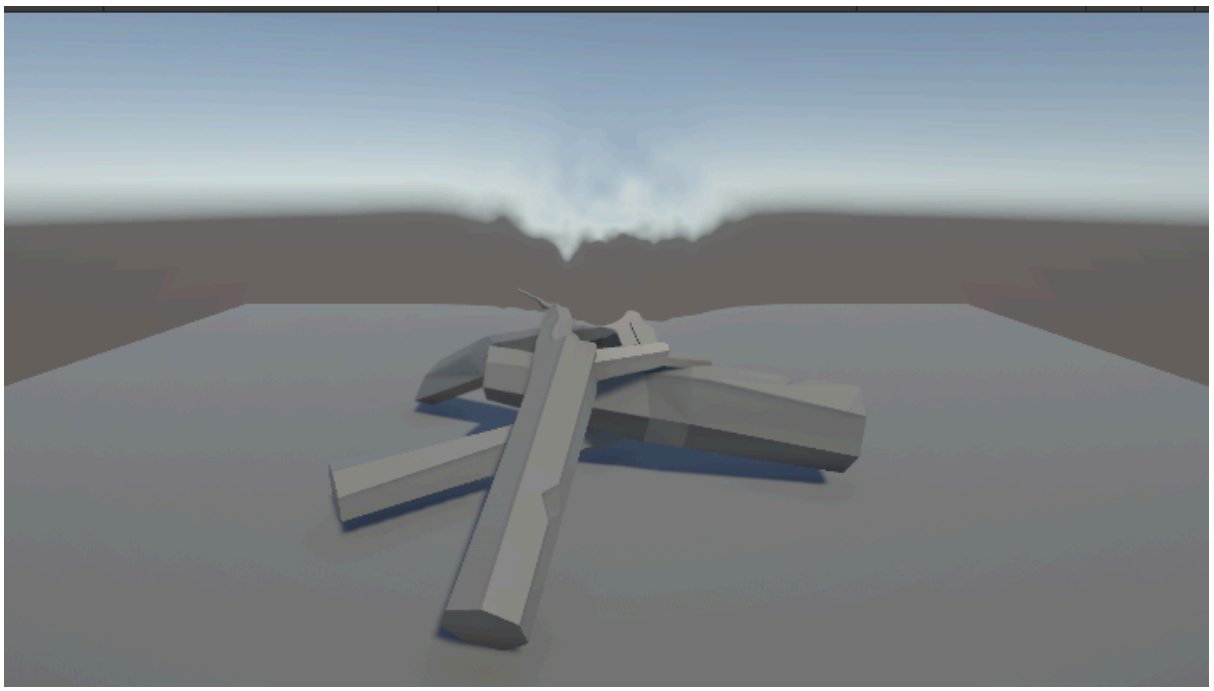
This shader and script uses material property block which enables each individual meshes to have same material but difference parameters which reduces draw call thus increasing performance.

Also shader has made with modular sub graph which makes you can integrate this in your shaders too , easily.



Heat distortion

A shader with plane that will distort the background and give the appearance of hot air moving . you need to have enabled opaque texture to work this. Put in the scene and adjust whatever you want in the shader setting and put in suitable places.



Heatable

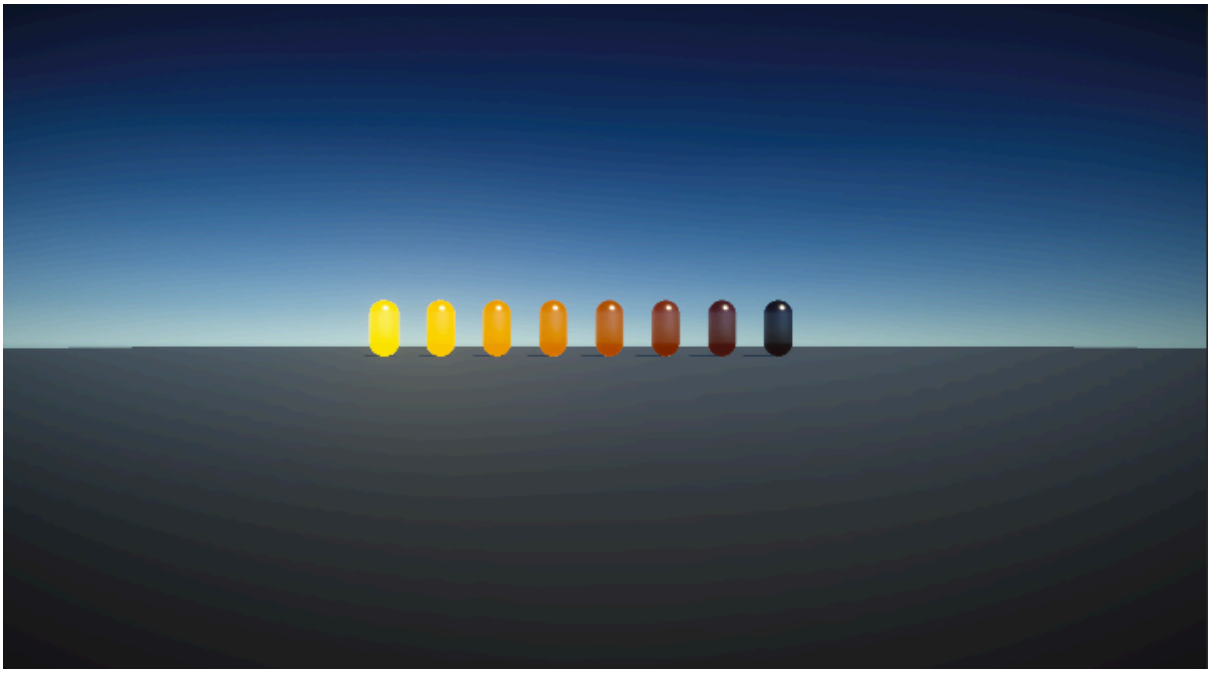
A physics script that will change material emission to simulate heating up , useful for physics simulations of metal , or to give the appearance of gun barrels heating up , feel free to edit the script however you want.

There are method to directly trigger heat up which is useful for giving the appearance of gun barrel heating up.
Heating up gradient is close to how real objects heat up.

Hover over the script properties and tooltips will reveal the more details.

Press play and wait until capsules fall to ground and see the asset in action

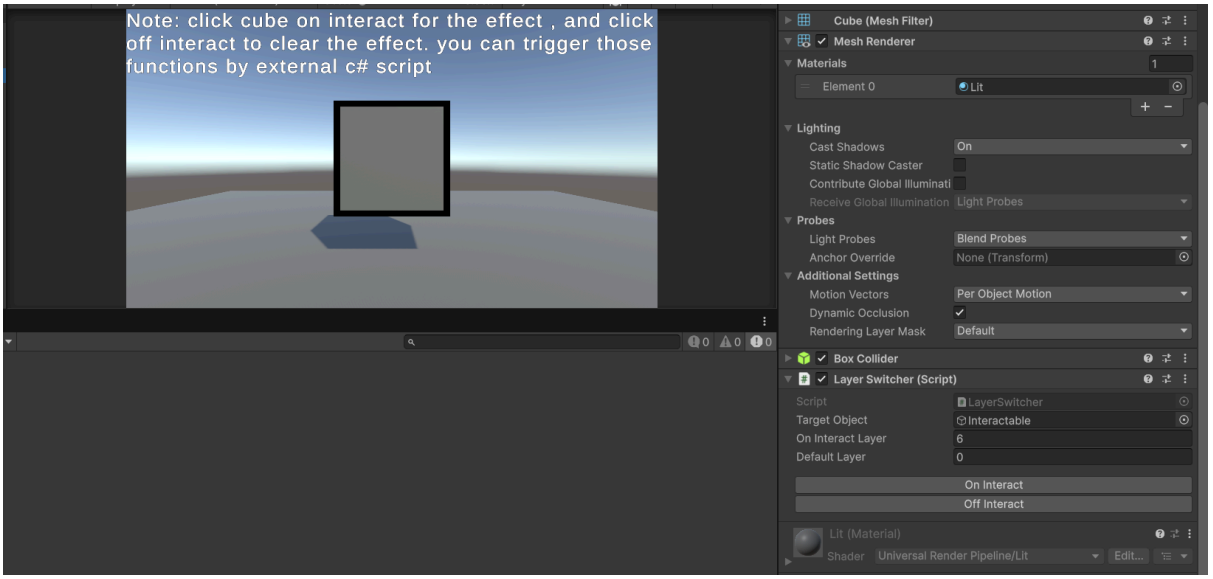
Tip - Heat cooling is faster when it is hot too much



Interactable (outline)

A out line shader with script that can tun it on or off .
To get started create a new layer (example outline) , put that layer on outline universal render data script render object pass.

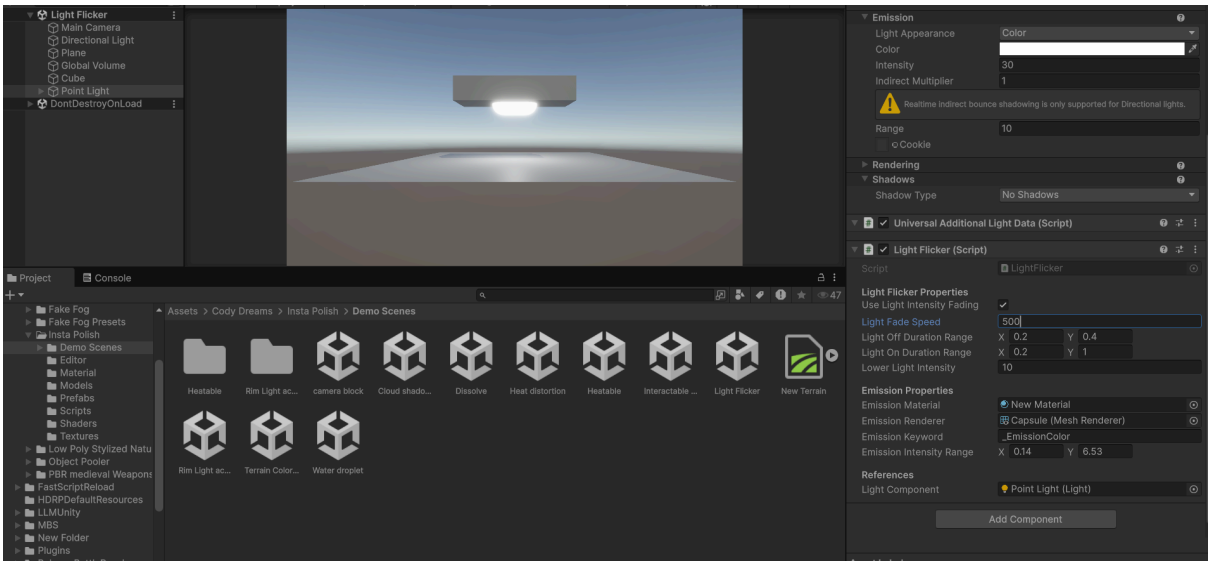
filters>Layer - newly created layer.
Then select that render data and put into a renderer and make it default or whatever your scene camera set to render and change models layer to that outline layer.
(you can use the given script to do that)



Light Flicker

A light clicker script that will simply flicker light randomly with random time frames .

You can use a material with emission on here , also you don’t need multiple materials and you can re use the material.
Select the light from the light flicker demo scene and configure your own parameters as needed. Hover over the parameters and tooltip will reveal more data about the parameter.

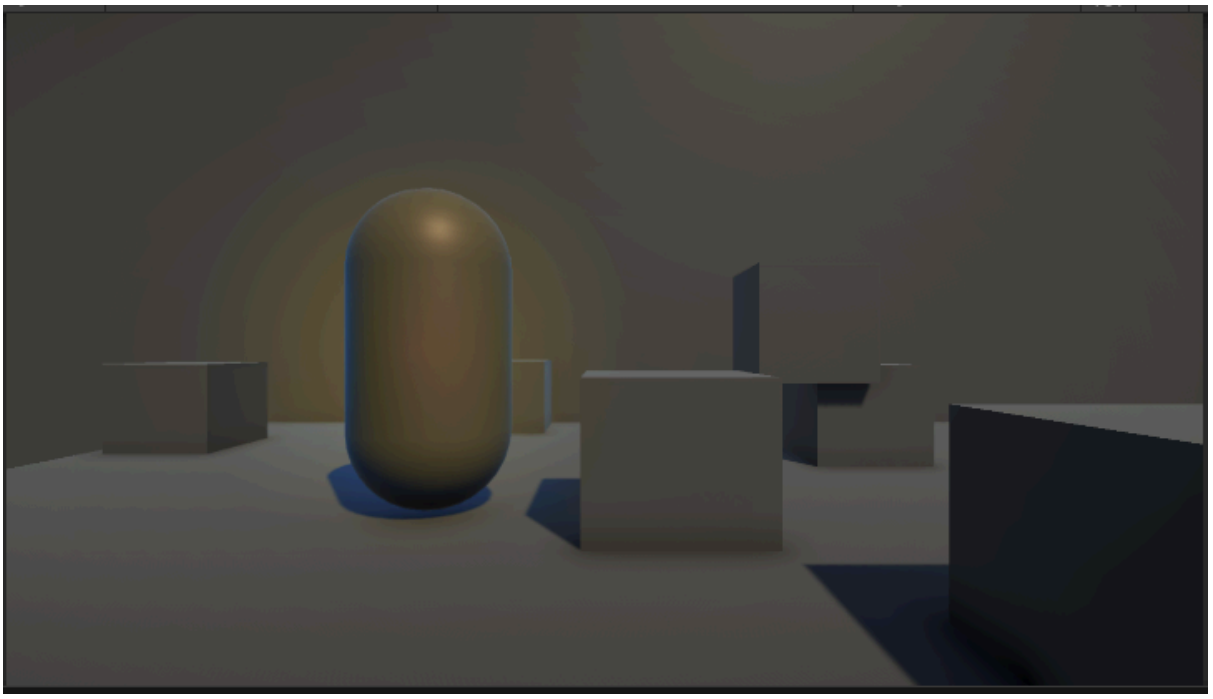


Rim light accent

A shader that will boost behind exposure and saturation by desired amount so subject could with stand against the rest of background and can draw attention , useful to bring attention and build cinematic games

Required opaque texture..

Saturation and exposure is bit exaggerated here to show effect clearly



Terrain Color map

A demo scene with script attach to camera that is placed above the terrain and capture a terrain color texture image. This image can be used to blend terrain color into objects such as grass (grass shader is not included , but a configurable sub graph containing correct shader functions are included).

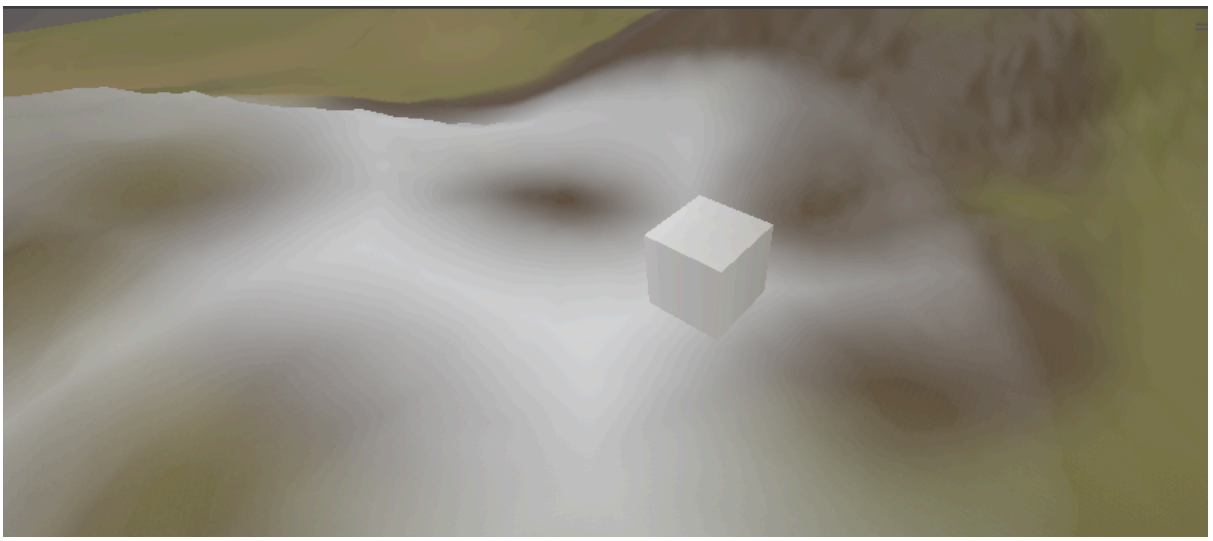
Check out the script attach to camera and configure the settings according to terrain and your requirements.

Important consideration , make the camera orthographic and set the terrain size into half of the actual terrain size . also set a high enough position to avoid capping out the mountains. Also don’t make the texture too big or it would be more performance expensive and it would have too many details (we just need pure color texture not detailed terrain texture).

Select the cube in the scene and move it in different places on the terrain where terrain texture color is different it should be reflected in the cube color.

In the shader put the offset into 0.5 (for most use cases).

Below image cube color is being white because underline terrain color is generally white.



Water droplet

A lens water dripping shader with a script that will react to camera movement dynamically.

To get started , make the outline render data into renderer and make it default . then go to outline renderer data and turn on the lens rain dripping feature. (we are re using same render data asset because we would not want to make a mess in your project plus making this testing and setuping this more easy)

Play the scene and move and rotate the camera and see screen water will react to those movements .
Moving forward will increase the water intensity while making the dripping down water go upward (simulating wind hitting the camera and make water droplets go up).
Going back will decrease the intensity because fewer and fewer water will hit the screen.

We recommend you may turn off the horizontal camera movement and rotation reaction towards water dripping due to the unnaturalness(via script editing or shader editing , currently we do not have implemented an option for that) .

Note currently rain dripping down is currently managed by the script so without the script rain drips won’t go down , so make sure to put it on the scene (if you building a new one).

Tool Tips will reveal more information about the parameter. Play and check out the shader and you will able to fine tune it as you want.