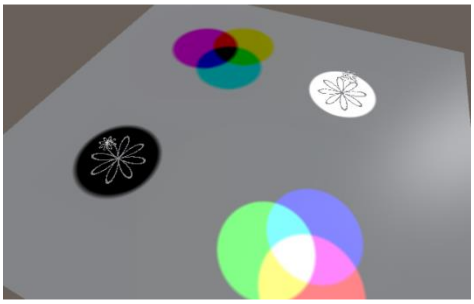
Negative Light

Introduction:

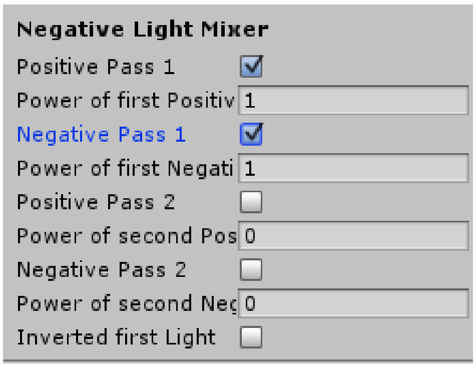
Negative Light as the name suggests is the opposite of normal light. Instead of lighting things up, it darkens the surrounding instead. Currently no one has observed such light in real life, however such light could exist in alternate universes.



The right image shows the negative lights on the left side and positive ones on the right. Negative lights can also combine and form color combinations. The color spectrum will consist of yellow, cyan and magenta instead of red, green and blue circles. Negative lights can also cast “negative” shadows. When positive and negative lights clash together, depending on the material settings, one type of light will always come out of the top due to shader limitations.

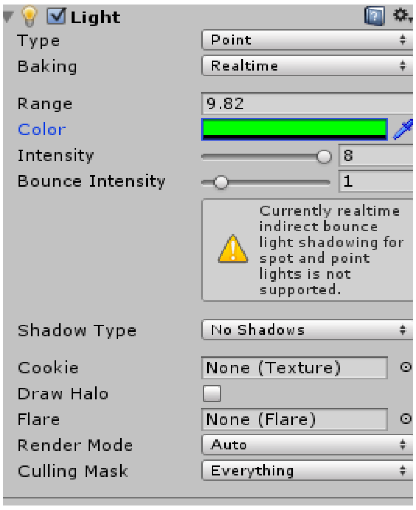
How to use:

This asset currently contains 3 shaders where two of them are enhanced versions of the standard shaders. For fast testing, select one of your materials which use the standard PBR shaders or create a new one and change it to: “NegativeLight/Standard/NegativeLight_Standard_X4” or the specular version depending on your workflow. The inspector is the same like the standard shader with the addition of the Light Mixer. At first no light will be visible because every light pass is disabled. To activate the lights, hit the Positive Pass 1 and Negative Pass 1 to get basic positive and negative lights. Whenever positive lights and negative lights clash together, these settings will determine the outcome.



Negative Light Mixer

Positive Pass 1	☑
Power of first Positiv	1
Negative Pass 1	☑
Power of first Negati	1
Positive Pass 2	□
Power of second Pos	0
Negative Pass 2	□
Power of second Neç	0
Inverted first Light	□



☑ **Light**

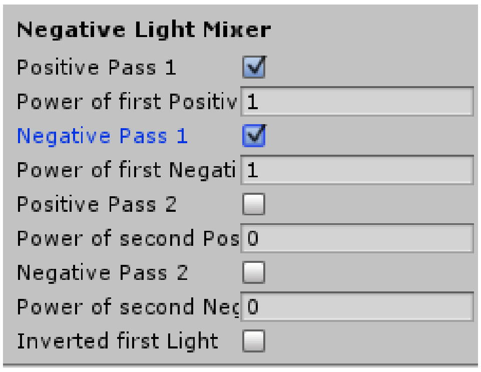
Type	Point
Baking	Realtime
Range	9.82
Color	🟩
Intensity	8
Bounce Intensity	1
	⚠ Currently realtime indirect bounce light shadowing for spot and point lights is not supported.
Shadow Type	No Shadows
Cookie	None (Texture)
Draw Halo	□
Flare	None (Flare)
Render Mode	Auto
Culling Mask	Everything

Next place normal lights into your world. Currently every light type is supported except **baked** ones because Unity will always generate a positive light map.

Now to make the light negative, set the alpha value of the color to zero. The other RGB values describe which color should be extracted out of the material. All other values like Intensity and the shadow settings work as usual like the positive light settings. However some values will generate unwanted results like Draw Halo and Flare settings because these are always positive.

Important Note: The Intensity value is multiplied with the color value which means that intensities below 1 will result into alpha values below 1 which can turn a positive light into a unwanted negative one. For weak lights, set Intensity to 1 and lower the RGB values instead.

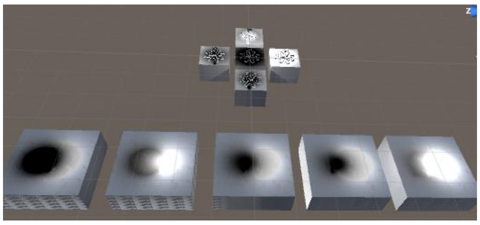
Light Mixer:



Negative Light Mixer

Positive Pass 1 ☑
Power of first Positiv 1
Negative Pass 1 ☑
Power of first Negati 1
Positive Pass 2 □
Power of second Pos 0
Negative Pass 2 □
Power of second Neg 0
Inverted first Light □

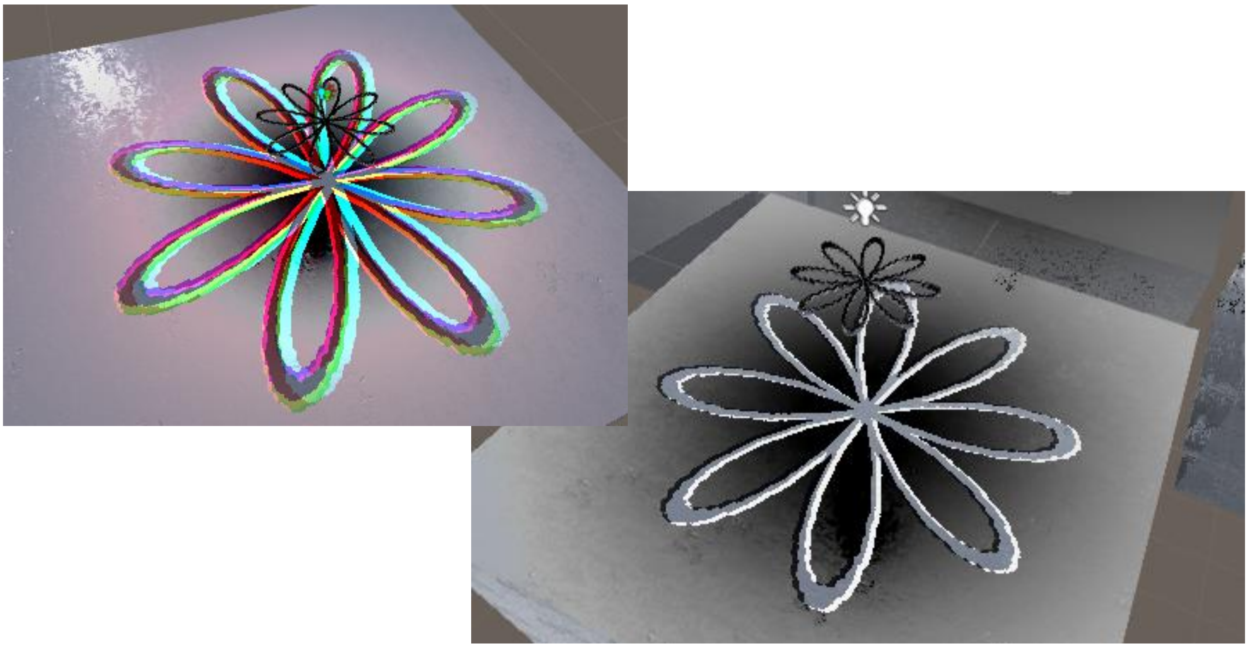
The light mixer is the most important component which describes how negative and positive lights will be rendered and what should happen when both sides collide. Every shader in this asset has these options. Each pass does its own light calculation and is multiplied by its according power value. Keep in mind that each pass has an own light calculation and costs performance. The standard mixer version X4 has four options. This means that lights are calculated 4 times which costs performance. The X2 shader versions have only 2 options (one negative and one positive) and just have the additional light calculation pass for negative lights. However the downside of the X2 versions is that the negative light will always win against positive ones.



The last setting "Inverted first Light" inverts the first directional light if the alpha value is zero. However keep in mind that the alpha value of directional lights is also influenced by the Intensity value. You can have positive and negative directional lights at the same time which can create bizarre results.

If shadows are enabled on both lights, the resulting shadow can end up becoming some bump mapping like shadow if both lights are very close to each other. When such lights are colored and more than two lights are involved, the result can be very unpredictable like in the left image below where two positive and two negative lights with different colors are placed very close to each other and a test 3D model which blocks the light.

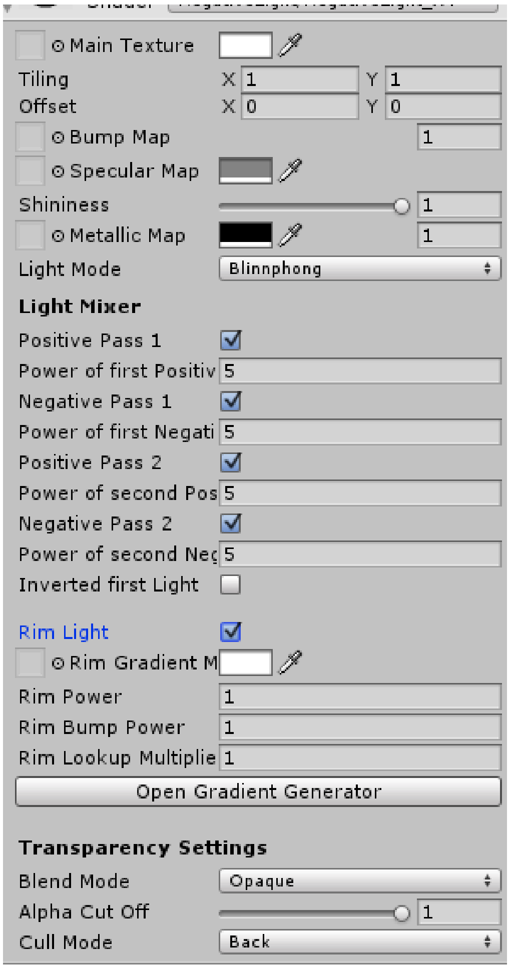
If you don't see all lights in the sample scenes, increase the maximum light count in the quality settings.



Custom Inverted Light Shader:

This asset also contains a custom shader which is has most common features of all legacy shaders combined into one. You can select one of the three common light models, transparence settings and cull settings additionally to rim lighting features. The rim lighting requires a gradient texture which is a simple texture that is only one pixel wide.

You can either use the preset gradient texture or generate one using the included gradient generator which is described below.



Main Texture
Tiling X 1 Y 1
Offset X 0 Y 0
Bump Map 1
Specular Map
Shininess 1
Metallic Map 1
Light Mode Blinnphong

Light Mixer
Positive Pass 1 ☑
Power of first Positiv 5
Negative Pass 1 ☑
Power of first Negati 5
Positive Pass 2 ☑
Power of second Pos 5
Negative Pass 2 ☑
Power of second Neg 5
Inverted first Light □

Rim Light ☑
Rim Gradient M
Rim Power 1
Rim Bump Power 1
Rim Lookup Multiplie 1

Transparency Settings
Blend Mode Opaque
Alpha Cut Off 1
Cull Mode Back

Parameters:

- **Main Texture:** The main texture with color multiplier and tiling options included
- **Bump Map:** Standard bump map with multiplier
- **Specular Map:** Specular map with color multiplier
- **Shininess:** Specular shininess
- **Metallic Map:** Map for cubemap reflection from reflection probe with color multiplier and power included
- **Light Mode:** Set the light mode algorithm. Has the common one which are Lambert, Blinn, Blinnphong
- **Light Mixer:** Section to set positive and negative light
- **Rim Light:** Section for rim light. Is enabled if checked.
- **Rim Gradient map:** Gradient map required for rim lighting
- **Rim Power:** Power of rim light
- **Rim Bump Power:** How much does the bump map influence the rim lighting
- **Rim Lookup Multiplier:** Change the lookup intervals. Recommended to keep at 1 unless funny results are wanted
- **Open Gradient Generator:** Opens the “GradientToTexture” editor
- **Blend Mode:** Common Blend modes which are Opaque, Cutout and Fade
- **Alpha Cutoff:** Alpha cutoff used when Blend Mode is set to cutout
- **Cull Mode:** Option to alter the culling mode.

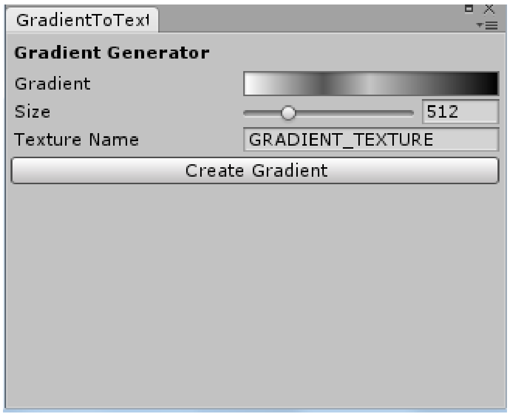
Additional Note:

This shader requires minimum shader model 3.0. If your device does not support 3.0 or higher, stick with the standard shader as it is implemented to support 2.0

GradientToTexture:

For rim lighting, you need a gradient texture to get decent results. It is very unintuitive to switch to your desired image creation tool so this little tool is added which simply transforms a color gradient into a texture so it can be used in shaders.

The tool is very simple to use. Just set your desired gradient with the standard gradient editor in unity. The size defines the length of the texture in pixels which is 512 by default. The texture name is simply the filename (type ending is added automatically).



Gradient Generator

Size 512

Texture Name GRADIENT_TEXTURE

When you hit the button, the texture file will be created and is located in the standard asset folder as a “PNG” file.

Important things:

Negative Lights are very unusual things so it usually not supported by game engines as they try to simulate real life graphics which has additive lights only. This package tries to implement negative lights as best as it is possible. However it is not possible to change the engine itself so some features simply won’t work.

- Baked Lights: Baked Lights will be bound into a texture and is done by unity internally. The resulting light map will contain red, green and blue values and therefore is additive.
- Halo: The standard light object has the halo setting which is additive. To obtain negative halo, create a billboard sprite object which always faces the camera and increases its size. However attempts to create looked terrible.
- Light Mixer: The light mixer allows power values greater than one. Recommended is a value of 0.5 which represents the default light strength. Keep in mind that no light will be visible when every pass is disabled.
- Mobile Application: These shaders require more processing power than those without negative light. Use the X2 versions as they only have the extra pass for the negative lights. However the runtime is still bigger although smaller than the X4 versions so use it only on those materials which should be influenced by negative lights.

Last Notes:

Every shader can be extended to support negative light. There are hundreds out there so I have included the most common ones used.

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: mhartl.mmt-b2013@fh-salzburg.ac.at

Homepage: http://lostinpixels.org/