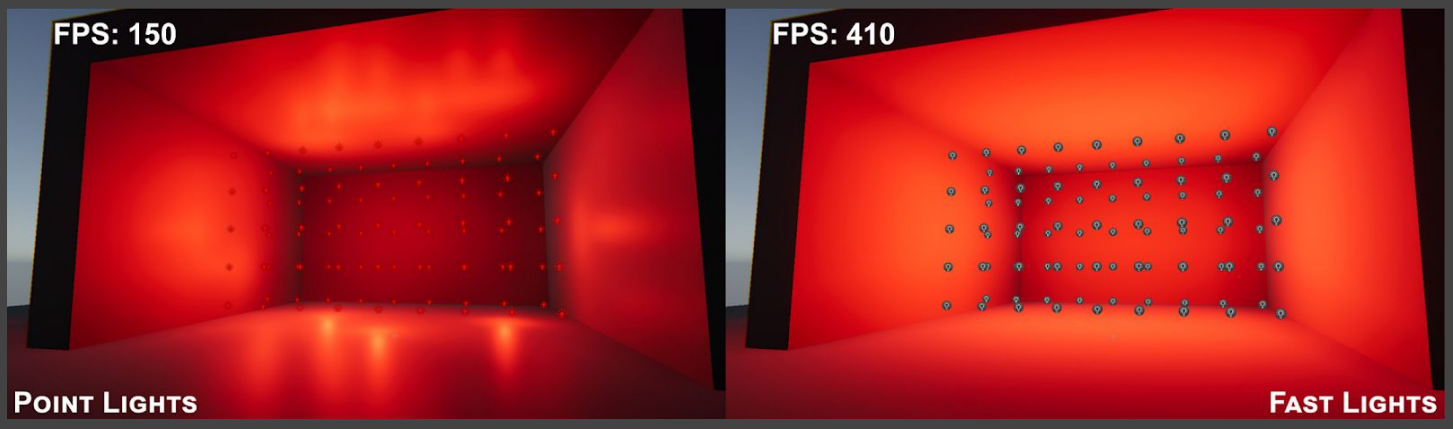
Developers Guide v1.0

Introduction

UPGEN Lighting is an optimized for maximum performance framework that aspires to provide a fully dynamic approximation of Global Illumination to your games (requires no precomputations). Initially designed for VR games, this solution provides high execution speed, produces great visual results, and uses a minimum of system resources. Emulation of indirect lighting could be made as fully automatic, in a semi-automatic way, or by manual designing light to save extra performance.

Under the hood, the solution is based on the idea of making an ultra-fast type of point light sources (10 times faster than standard ones) and using thousands of them to approximate light propagation for the scene in realtime. To achieve this we made a special kind of screen space effect, which bakes each frame hundreds of most valuable lights and applies combined lighting inside G-Buffer for the current camera.



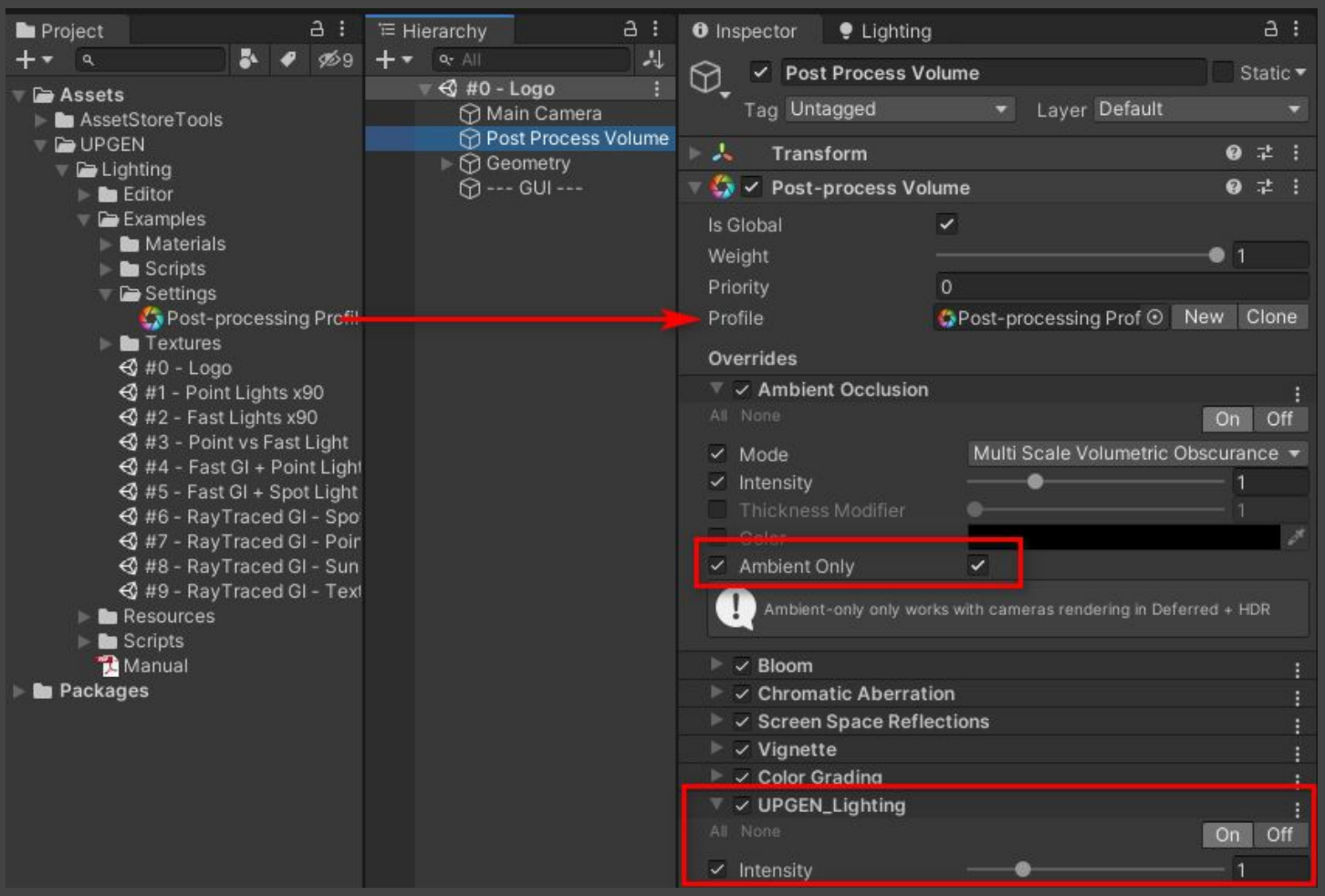
This framework provides several workflows for building lighting in your scenes. You can use such Fast Light sources to automatically transfer bounced light from standard light sources using Ray-Tracing. Or you can combine a standard type of light sources with Fast Lights to manually approximate GI in complex scenes and achieve the best performance and a beautiful visual look.

Installation

Step 1. Go to AssetStore, download, and import **UPGEN Lighting** package. The first time you should use a temporary or an empty project for installation to guaranty that you will not break your main project files or settings.

Step 2. Create Post Processing Volume and add Post Processing Layer component to main camera. For Post Processing Volume you can create own profile or just use one from examples folder:

Assets \ UPGEN \ Lighting \ Examples \ Settings \ Post-processing Profile.asset



Step 3. If you are using «Ambient Occlusion» post effect to you should set its property «Ambient Only» to True.

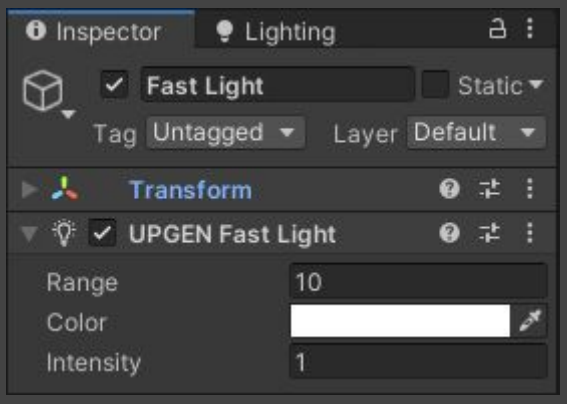
Step 4. Add «UPGEN_Lighting» post effect to the profile of some global volume in your scene and set effect intensity to 1.

Step 5. Ensure that you are using Linear Color Space. To do this go to Project Settings in the «Player» category, «Other Settings» section, «Color Space» property.

Step 6. Ensure that you are using Deferred Shading rendering path.

Fast Light

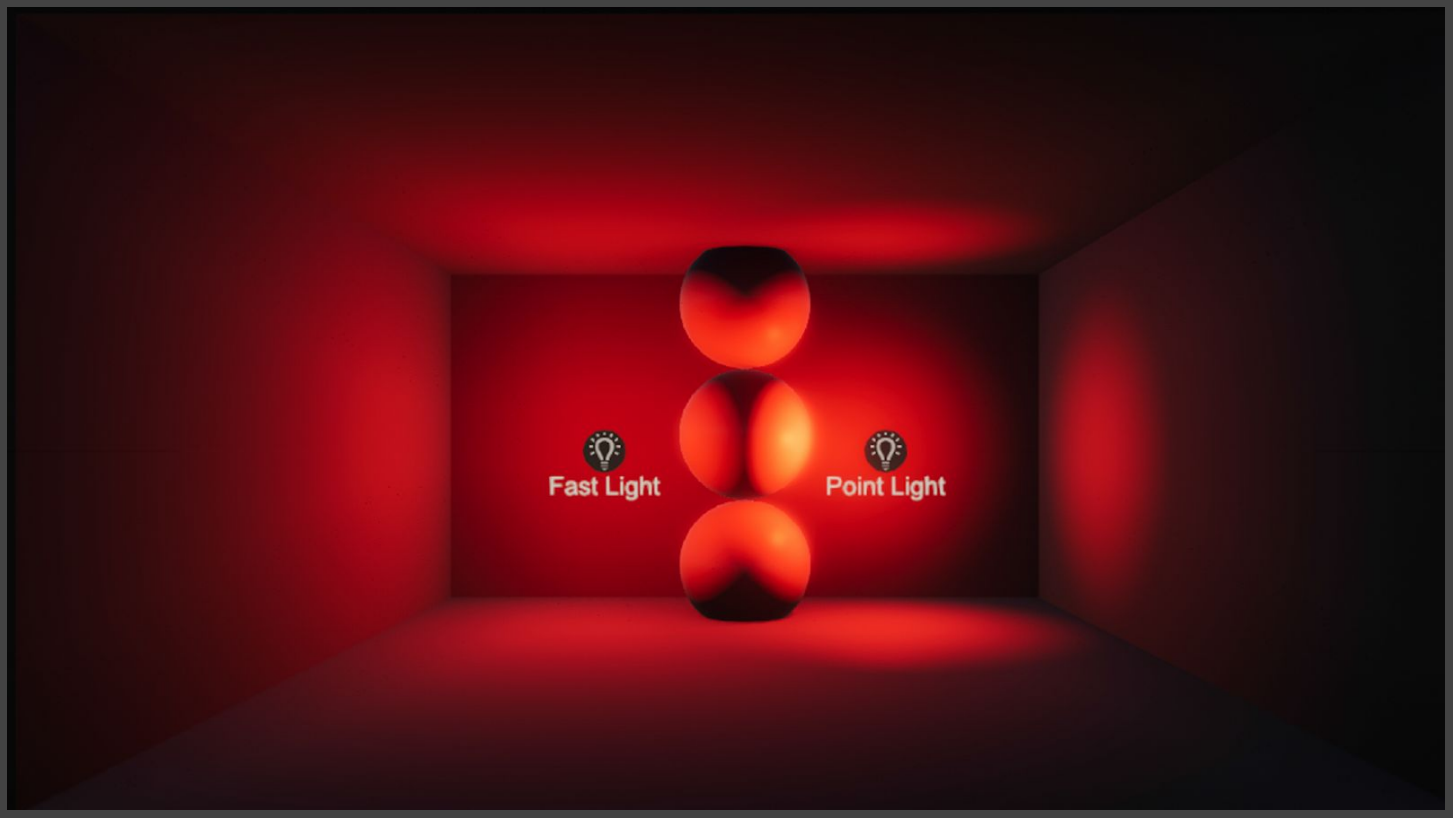
Fast Light. It produces a single light source for post processing and is designed to be a cheap alternative to Point Light. That is why all its properties are similar to base one.



*** Color.** Just a color of the emitted light multiplied with the temperature.

*** Intensity.** Set the brightness of the light. Multiplied by color it gives the final lighting amount.

*** Range.** Define how far the light emitted from the center of the object.



Fast Light - Best Practices

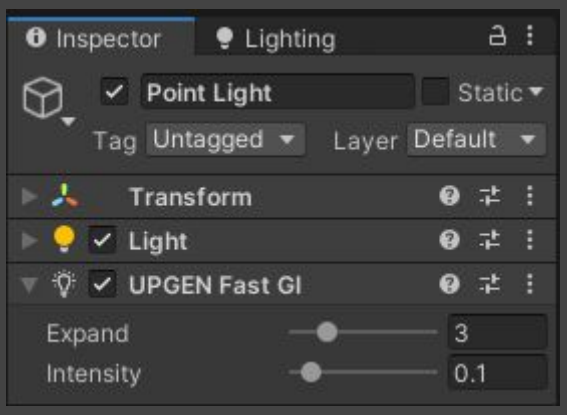
Rule - any emission should be reinforced by Fast Light.

As far as we used this framework in VR games, we pushed it to the limit to achieve maximum immersion. Our main rule is simple - even every tiny emission source like a light-emitting diode that's present on the texture of an object should emit real light. So when the player put his/her hand or any other object near to this light source he/she should see the lighting coming from it. So literally every texture part of the object with emission or small emissive objects could / should have corresponding Fast Lights.

Take a look at this facade of Santa's House (see the image below). Each small Christmas light here has its own Fast Light so we could have hundreds of them in a single frame when the player is looking on the facade. However, he/she can come close, inspect each one, and see that all these lights are really emitting light, and are not just present as emissive materials.

Fast GI

Fast GI. It is an addition to the basic light component. It takes all its properties to add bigger Fast Light around which it will fake indirect lighting. It supports only Point and Spot types of basic light and does nothing for Directional and Area lights.



*** Expand.** Multiplier - represents how big Fast Light will be (around Point Light or the first part of the cone for Spot Light). Value 1 means range equals to base light range. Value 3 means 300% or 3 times bigger range than base light has.

*** Intensity.** Multiplier - represents a percent of intensity that will be taken by Fast Light from basic light. Value 1 means 100% or the same intensity as base light has.

Fast GI - Best Practices

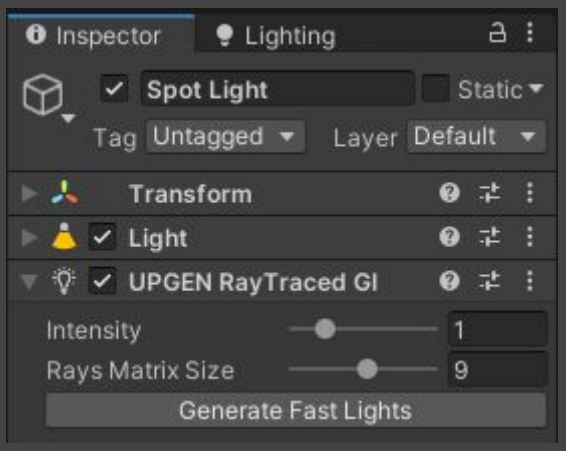
Rule - Point Light should go along with Fast Light.

Any small lamp or a candle should have a small (1-2 meters) and intense Point/Spot Light to make a shiny specular highlight on the objects in a small radius around and it should also have a big Fast Light, which will fake indirect lighting from it in the whole room. Fast Light should be 3-4 times bigger in range than basic Point Light and 10-15 times less intensive.

This approach could be easily implemented by attaching Fast GI component to your basic light. However, you could create two light sources separately - first with Point/Spot Light somewhere in the room and second with Fast Light closer to the center of this room to fill it with bounced ambient light. In this case, make sure to find such a range for Fast Light, which will make a visible gradient of intensity falloff from the center of the room to its corners (so corners could stay dark enough).

Ray-Traced GI

Ray-Traced GI. It is an addition to the basic light component. It takes all its properties to add many Fast Lights on the ends of bounced rays. To build bounced rays is creates a matrix of directions depending on the light type. For Spot Lights, it builds a regular grid of rays inside its cone. For Point Lights, it builds a spherical grid of rays to cover all its directions. For Directional Lights, it builds a regular grid of parallel rays around the pivot of light position.



*** Intensity.** Multiplier - represents a percent of intensity that will be taken by bounced Fast Lights from the basic light source.

*** Rays Matrix Size.** This defines how many rays will be used in the matrix [N*N] to approximate the indirect lighting of this light source. E.g. for size 9 it will use 81 rays, for size 5 it will use 25 rays. In case if ray hit something, it will produce bounced Fast Light.

Ray-Traced GI - Best Practices

Rule - Use realtime Ray-Tracing, remember the result, fake it manually.

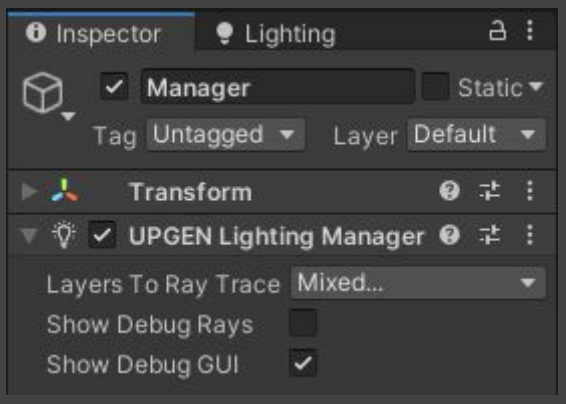
When we want to gain extra performance, we could do such design trick. First, we put highly detailed light with a crazy amount of rays for a better understanding of how GI propagates in this room. Then we just disable Ray-Tracing and approximate final result by the manual placing of Fast Lights.

Rule - Use Ray-Tracing GI with Fast GI together for important lights.

Especially for Spot Lights it makes sense to combine both methods to get nice GI around starting part of the light cone with Fast GI and beautiful GI from the rest of the cone with Ray-Traced GI. Just try this with any lamps hanging on the ceiling. You will get beautiful light attenuation around the lamp on the floor from Fast GI and nice bouncing in the rest of the room from Ray-Tracing GI.

Manager

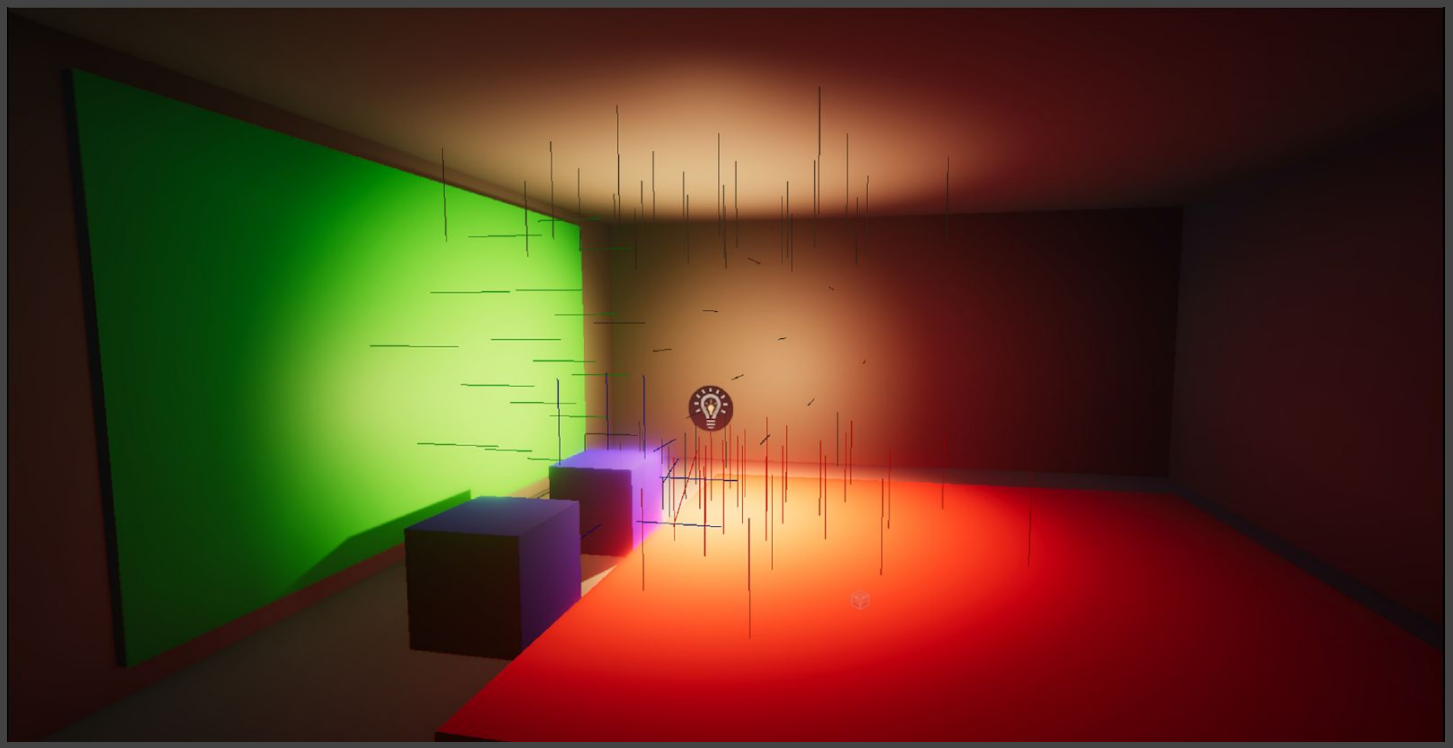
Manager. This is a global lighting preferences holder in the scene. It should be only one per scene. If absent - default values of all its properties will be used.



*** Layers to Ray-Trace.** Defines which layers would be used and ignored by a realtime ray-tracing. Usually, you need to ignore Player, Water, Inventory, UI, Triggers, and other types of special colliders.

*** Show Debug Rays.** Enables to see special lines, which represent bounced rays with Fast Lights on their end (in the Scene View).

*** Show Debug GUI.** Enables to see in top-left corner statistics about lights count of each type and Fast Lights capacity in post processing.



Limitations

- Only Built-in pipeline with Deferred Shading is supported (does not work with HDRP or URP pipeline)
- In automatic mode it could be very inaccurate (to stay fast) and some manual work still required
- Lighting does not affect forward-rendered objects like transparent geometry
- Ray-Tracing uses colliders to detect ray hits (geometry without colliders will not produce bounced lighting)
- Best light types to use with ray-tracing are Point and Spot Lights
- Area type of lights is not supported by a realtime Ray-Tracing
- For Directional Lights a realtime Ray-Tracing works in a small area around its pivot and you should design indirect lighting for big outdoor scenes in semi-automatic mode
- Does not take into consideration light cookies
- Ray-Tracing cache has a certain amount of inertia and will have visual delay for fast-moving light sources
- Ray-Tracing will produce light leaking

F.A.Q.

What platforms does it support? Can I use it for VR?

It should work on any platforms. RTX video card is not required to use Ray-Tracing components. All types of VR HMDs are also supported.

How to change the limit of Max Lights Count from 96?

You need to change constant «MAX_LIGHTS_COUNT» in two places - in the script (***UL_Renderer.cs***) and in the shader (***UPGEN LIghting.shader***). However, be careful - making it too high can affect performance, try to keep this limit as low as possible.

How would this work for procedural levels? First/third person cameras?

It is fully runtime, so it is a perfect solution for such cases. It does just what you are expecting if you provide colliders to your geometry. As for cameras - you can use any kind of view you want.

Feedback & Support

If you need help with UPGEN Lighting or would like to share your feedback and experience or want to report an issue - you can visit the thread on the Unity Forums or you can email me at **anton.tril@gmail.com**.

If you are reporting an issue, please provide a detailed description of the problem you are experiencing, screenshots, and system information (graphics card, operating system, Unity version, etc.).

Standard
UPGEN
LIGHTING

Changes Log

v1.4

- Added Ray-Traced GI rays smoothing for fast moved Fast Lights
- Added Unity 2018 support

v1.3

- Added main menu commands to create Fast Light and Manager objects
- Updated script icon for Manager script

v1.2

- Ray-Traced GI: when used with Directional Light new property “Rays Matrix Scale” can adjust scale of ray-tracing matrix (how big area should be covered by rays)

v1.1

- Ray-Traced GI: new button to Create/Destroy set of Fast Lights from internal rays
- Examples: hide next/prev buttons if you have no scenes in build settings
- Manual: quality improvements + size reduced