Welcome To 2DVLS!

Thank you for purchasing the 2D Volumetric Lighting System from Reverie Interactive. This document has been written in an attempt to help you understand how the 2D Volumetric Lighting System is used and how you can make the best out of this product.

In the following sections we will be covering how to setup a simple light, how to edit the properties of that light, how to improve performance of the light using the new systems in 2DVLS, and finally how your light can be linked up to other objects via the powerful event system which is built in to 2DVLS.

I hope you enjoy this product and find it helpful in your development adventures!

-Jake Fletcher

Tutorial 1 - Simple Light Setup

Setting up new lights in 2DVLS is very simple and only involves 3 clicks. Once you have installed the package go to “GameObject” -> “Create Other” -> “2D Light”. You will see a new 2d light appear labeled “Created Light”. You can rename this now if you would like.

If an error occurred during the creation of your light at this point then it could be that you deleted the “Resources” folder located inside of the “Light2D/Core” directory. This folder is used to load default materials into your lights at creation. If you do not mind a few errors then go ahead and delete this folder and make sure you assign a new material to your light. Otherwise, please leave all fodders within the “Core” directory intact.

Tutorial 2 – Altering the Lights Properties

Okay so now that you have your new light it is time to understand and change some of its properties!

Ignore Optimizations: The first property you notice is an “Ignore Optimizations” checkbox. This label is slightly misleading as it doesn’t actually ignore ALL optimizations. What this checkbox does is allows your light to update on every “LateUpdate()” iteration regardless of if there is an object in the lights view or not. This could be useful if your scene had a bunch of static objects in it (we will cover this later) and you wanted to move this light to be dynamically moved or to a new location.

For example: let’s say you’re using the light as a FOV game mechanic on 50 zombies. To increase performance you would only want the light to update when it’s being moved otherwise you would want the light to just have “baked” settings. Well to do this you could set this checkbox to true while the zombie is moving and when it stops set it back to false.

Use Events: The next property you might notice is labeled “Use Events”. This checkbox might not need much explanation so basically it is another optimization feature. If you have “Use Events” checked the light will run through all of the processes needed to collect changes in the scene nearby. This can get memory intensive as it requires 2 lists of objects. So when you do not need the light to trigger events, its best to leave this setting to false.

Light Radius: This setting allows you to set the size of your lights.

Light Color: This setting allows you to set the color of your lights.

Light Material: This is used to set the material of your 2D light. In previous versions of 2DVLS this was very limited as to what you could use as a material. In this version of 2DVLS you're finally able to use practically any material you want! By default the material that is used is the "Mobile/Particles/Additive" to insure that the system works on mobile devices and since it gives in my opinion the best effect. You might also notice that UVS now work correctly no matter what orientation you have your lights in!

Sweep Start: This setting is used in conjunction with "Sweep Size". If your using a spotlight configuration then basically this function allows you to set the direction that cone is pointing.

Sweep Size: This setting sets the cone angle of your lights. The default setting is a full 360 degrees but when using the light in a spotlight configuration you would want this to be more like 45 degrees.

Light Detail: This is a new feature to 2DVLS. In previous versions I gave you the freedom of manually setting the detail to a range of 4-Infinity. I received some comments suggesting that this was too cumbersome and that people didn't really want to deal with that system. The reason a detail setting is required is because 2DVLS doesn't know what scale your scene objects are in and every scene is created differently. For this reason I could not completely remove the detail setting. However, I am able to drastically simplify it!

The settings available to you are: Low, Normal, Medium, High, Very High, and Extreme. You should not use the Very High or Extreme values when you are dealing with multiple lights as it calls Physics.Raycast 3001 and 5001 time respectively which will drain your processing power. The default setting is Normal which emits 501 rays per LateUpdate() when an object is inside the lights sphere.

Shadow Layer: This is the easier to use shadow layer system. In the previous version of 2DVLS this whole thing was very cumbersome to use and caused a lot of headaches. With the new system it works exactly the same as everything else layer based in unity. Basically if you want only object with a "Shadow" layer assigned to them to render a shadow, then you would set this value to "Shadow". If you want everything that has a collider to cast a shadow then set this to "Everything". "Everything" is the default setting when you add a new light via the menu.

Tutorial 3 – Improving Performance

Improving the performance of your lights can be very important when you are dealing with a large number of 2d lights in your scenes. Thankfully, performance enhancements are easy to accomplish following a couple steps! First, if your light does not move then make sure the objects around it that do not move are set to "Static". This tells the light that even though it should cast a shadow off of that object, it will not have to update the light until a non-static object comes into range. If you're doing a top-down shooter make sure your floor is set to static as well since the detection area is spherical.

Tutorial 4 – Event System

Finally the most desired part of the 2DVLS system, Its Event System! In order to use 2DVLS in JavaScript you will need to create a "Plugins" folder in your root assets directory and move the Light2D.cs script into that folder. The reason being is script execution order. Light2D.cs must be executed before it can be used with JavaScript and placing it in the Plugins folder ensures this will happen.

Let say you want your light to change its color from red to blue when its rays hit a cube. To do this you will want to attach a script to the cube that links to your 2DVLS event system. Thankfully this is very simple via script!

First we need to create a new C# script and label it something like "BlueLightEvent.cs". Once you have done that go into the script. Before we link our script up to 2DVLS we will need to create a function that will interact with the light. This function MUST follow the following syntax:

```
void Function_Name(Light2D light, GameObject obj) { }
```

So let's make a function!

```
void OnLightEnter(Light2D light, GameObject obj)
{
	if(obj.GetInstanceID() == gameobject.GetInstanceID())
		light.lightColor = Color.Blue;
}
```

The reason we compare InstanceID() with one another is because we don't want one lights event triggering ALL of the functions in the scene.

After creating the above function we can now link it to our 2DVLS system by making a single call in the "Start()" function.

```
void Start()
{
	Light2D.RegisterEventListener(LightEventListenerType.OnEnter, OnLightEnter);
}
```

The "LightEventLIstenerType" enumeration has 3 values: "OnEnter", "OnStay", "OnExit" which correspond to the 3 events that are called.