LIGHT FLICKERING DOCUMENTATION

Thank you for purchasing this package

Light Flickering is a simple system that can make your lights flicker in a very easy and customizable manner, whether it's in random or manual mode. This will help you focus on your game and design rather than wasting time and programming something so minutely complex. It's already done for you!

GETTING STARTED

1. Add a light to your scene. Any type of light.
2. Add the *LightFlickering* script to the object.
3. Drag and drop the light component to the **LightSource** property.
4. Click on **RandomizeFlickerings**.
5. Make a new script (that will trigger the LightFLickering script when spacebar is pressed) and call it *SpacebarToFlicker* or anything you like.
6. Inside this new script:

```
using UnityEngine;

public class SpacebarToFlicker : MonoBehaviour
{
   public LightFlickering LF;

    // Update is called once per frame
    void Update()
    {
        if(Input.GetKeyDown(KeyCode.Space)){
            LF.flicker();
        }
    }
}

```

7. Add this new script to any object then drag and drop the *LightFlickering* script to the **LF** property of this new script.
8. Play the game, on pressing spacebar the light should randomly flicker on and off.

PROPERTIES AND METHODS

lightSource: the light component that you want to flicker.

randomizeFlickerings: set to true or false, depends on whether you want the flickerings to be random and non-stop. Randomizing flickerings generates a random time between two floats a minimum and a maximum.

minRandomizeTime: the minimum float to randomize a flickering time from between.

maxRandomizeTime: the maximum float to randomize a flickering time from between.

flickerings: an array of floats that takes in the amount of flickers you want as well as the timings.

loop: set to true or false, depends on whether you want the manual flickerings to finish after their first pass or to continue in an endless loop.

lightings: set the light color and bulb material of each flicker index. Giving you full control over color customizability. This will be read in both random and manual mode. If you want the colors to stay the same as the original light source simply set the size to 0. In random mode, when the flicker index is bigger than the length of this property, it will loop the colors from the beginning. In manual mode it'll stay put at the last color.

playBuzzSound: do you want a sound (AudioSource) to be played when the light is on. Adds more immersion.

buzzSound: the audiosource you want to be played when the light is on.

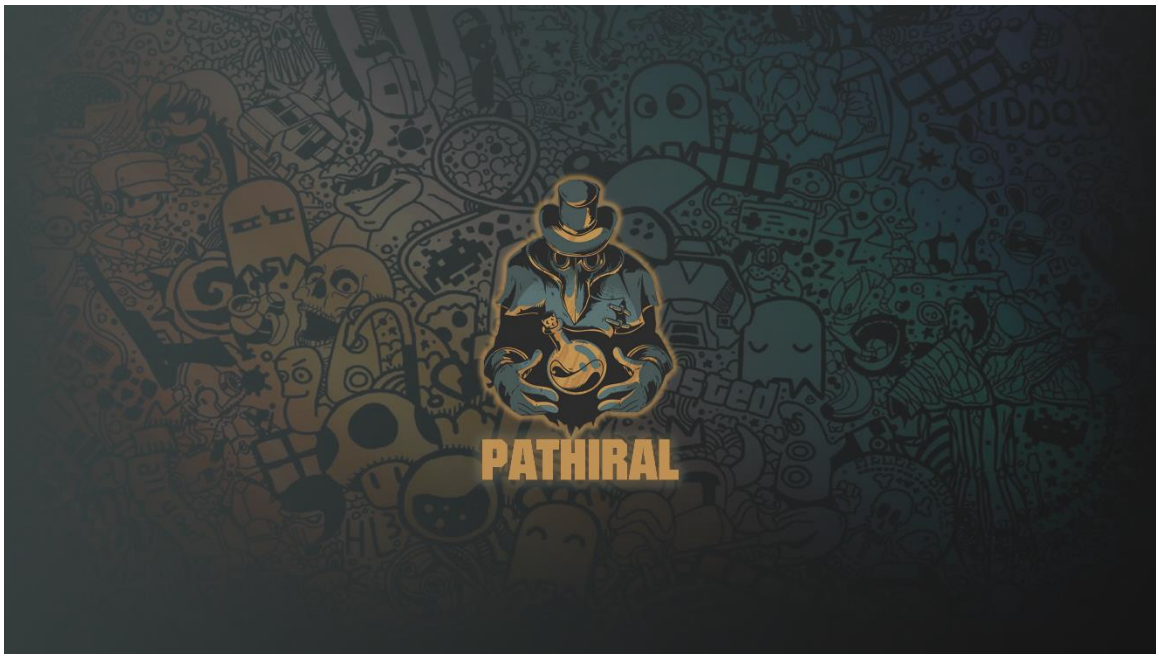
changeMaterial: do you want the material of a certain object to be changed when the light is off during flickering? Adds more realism as you can change dynamically the material of the light bulb to be dark to imply that the light bulb is off.

bulbObject: the object with the *MeshRenderer* that you want to change it's material.

newMaterial: the new material you want to replace the old material when light is off.

flicker() -> triggers the script and starts the flickering whether it's in manual or random mode depending on what you selected.

stopFlickering() -> stops the flickering.



Thank you and feel free to email me at *muradelboudy95@gmail.com* for support, questions or custom scripts.

More from ***Pathiral*** here