Day and Night Cycle

1. Create a new scene

Start by creating a new scene called “Day and Night”.

Create an Empty GameObject and call it “SceneManagement”

Add a 3D Cube named “Floor” and a 2D Capsule named “Player”.

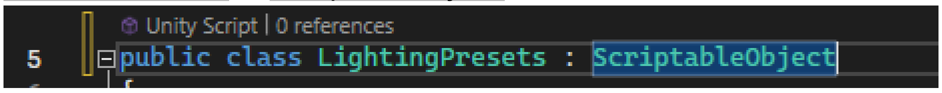
Make a new “Green Material” and drag it to the “Floor”, then make a “Blue Material” and drag it onto the “Player”.

2. Create Script

Create a script called “LightPresets”, this script will hold the light conditions of the scene.

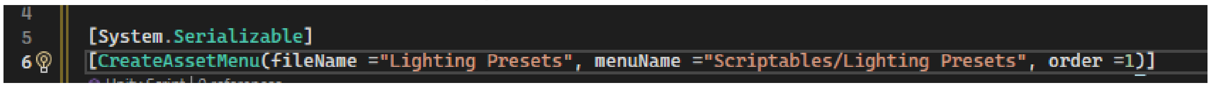
With this script we are going to be controlling the “Ambient Color”, the “Directional Light Color”, the “Rotation of the Directional Color” and the “Fog Color”.

As soon as we open this script we will make it a “ScriptableObject” by changing “MonoBehaviour” to “ScriptableObject” in the class line.



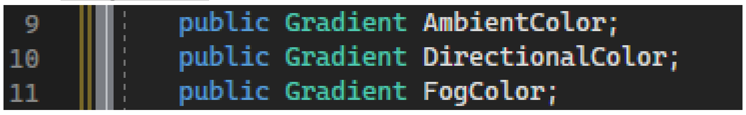
```
Unity Script | 0 references
public class LightingPresets : ScriptableObject
```

Next we will be adding a tribute on top of the class line like this:



```
[System.Serializable]
[CreateAssetMenu(fileName ="Lighting Presets", menuName ="Scriptables/Lighting Presets", order =1)]
```

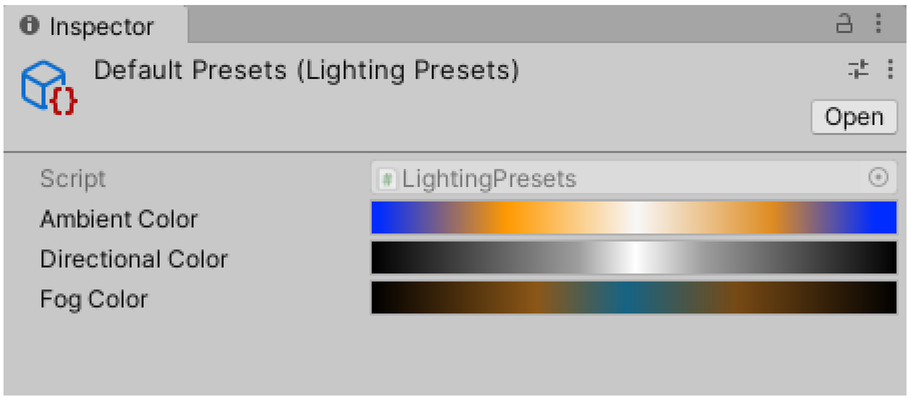
After this we will be creating Gradient variables for the “Ambient Color”, “Directional Color” and “Fog Color”



```
    public Gradient AmbientColor;
    public Gradient DirectionalColor;
    public Gradient FogColor;
```

3. Create the preset

After saving the “LightingPreset” script we will comeback to Unity and create a new Preset by right-clicking on the assets folder, Create>Scriptables>Lighting Presets. This new preset will be called “Default Presets”. When this is created we will proceed to crate the gradients for the three colors previously mentioned like this

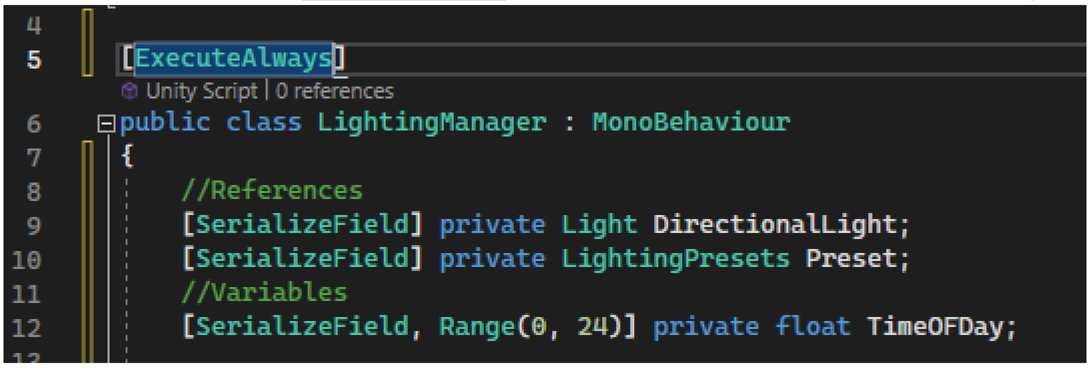


Script: LightingPresets
Ambient Color
Directional Color
Fog Color

4. **Create a new Script**

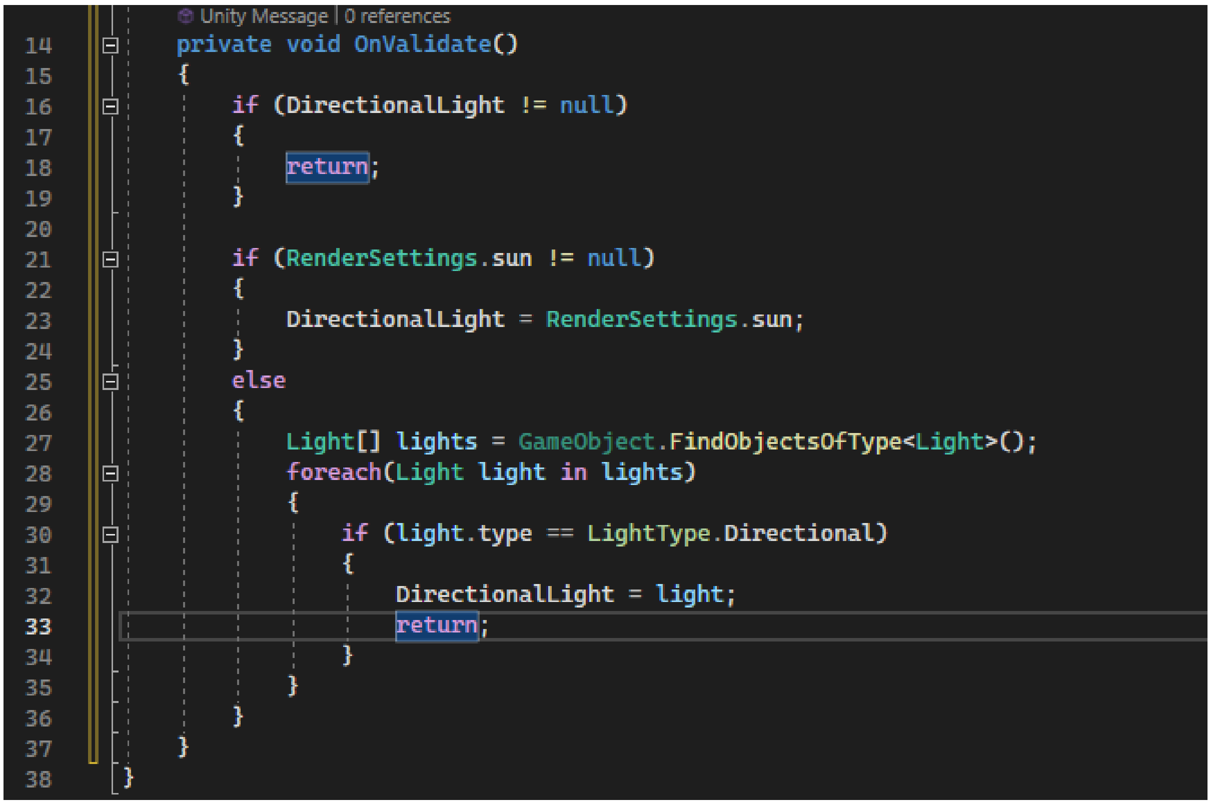
Create a script called "LightingManager", this will allows us to control the lighting as the day progresses.

As soon as we open this script we will add a tribute "ExecuteAlways" and some private variables which will be "SerializeField" so we can take a look at them on the inspector.



```
[ExecuteAlways]
Unity Script | 0 references
public class LightingManager : MonoBehaviour
{
    //References
    [SerializeField] private Light DirectionalLight;
    [SerializeField] private LightingPresets Preset;
    //Variables
    [SerializeField, Range(0, 24)] private float TimeOFDay;
```

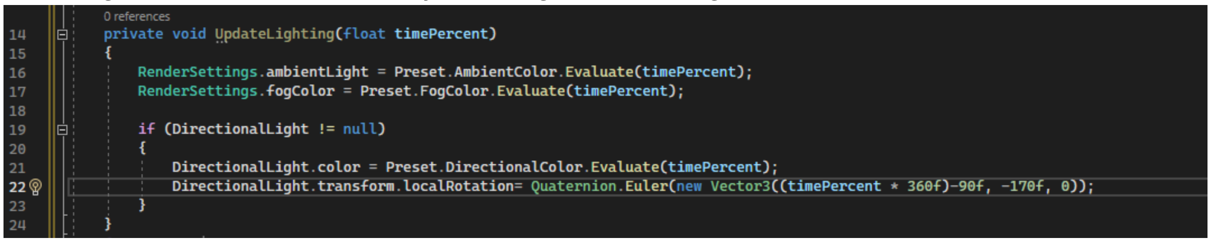
We are going to add a "OnValidate" function to the script and a few if statements just like this:



```
    Unity Message | 0 references
    private void OnValidate()
    {
        if (DirectionalLight != null)
        {
            return;
        }

        if (RenderSettings.sun != null)
        {
            DirectionalLight = RenderSettings.sun;
        }
        else
        {
            Light[] lights = GameObject.FindObjectsOfType<Light>();
            foreach(Light light in lights)
            {
                if (light.type == LightType.Directional)
                {
                    DirectionalLight = light;
                    return;
                }
            }
        }
    }
}
```

Next we will make another function that will help us change the lighting depending on the time in-game, this will be done by entering the following code:



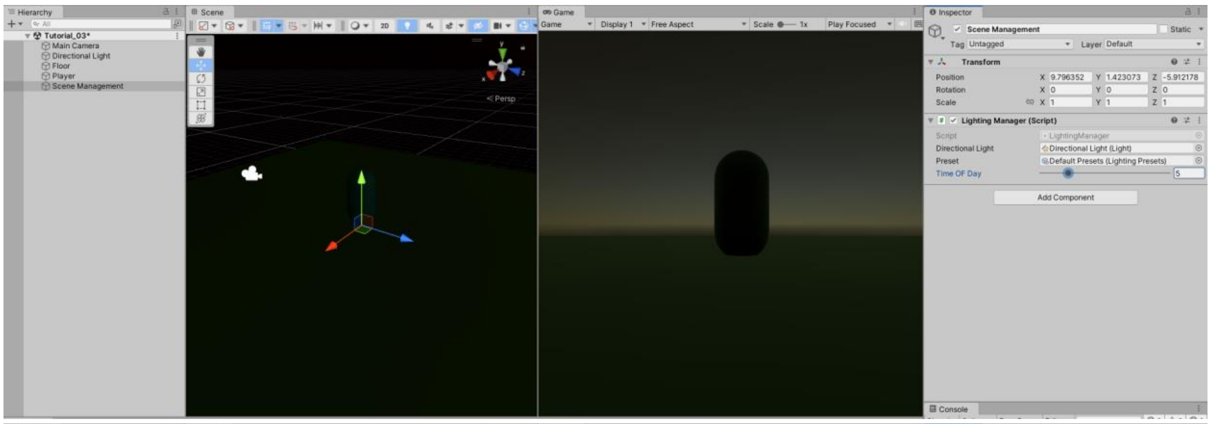
```
    0 references
    private void UpdateLighting(float timePercent)
    {
        RenderSettings.ambientLight = Preset.AmbientColor.Evaluate(timePercent);
        RenderSettings.fogColor = Preset.FogColor.Evaluate(timePercent);

        if (DirectionalLight != null)
        {
            DirectionalLight.color = Preset.DirectionalColor.Evaluate(timePercent);
            DirectionalLight.transform.localRotation= Quaternion.Euler(new Vector3((timePercent * 360f)-90f, -170f, 0));
        }
    }
```

Finally, we will use the "Update" function to check for a couple of things, the first one being if we have assigned the Presets. Secondly, we will update the lighting also through the use of the "Update" function. This will be executed by the following code:

5. Testing

Finally, we will add the "Lighting Manager" script to the "SceneManagement" game object we created before, and we will drag the "Default Presets" we created to the "Prest" slot, and we can use the slider to see if the scripts work.



Hierarchy
- Tutorial_03*
  - Main Camera
  - Directional Light
  - Floor
  - Player
  - Scene Management

Inspector

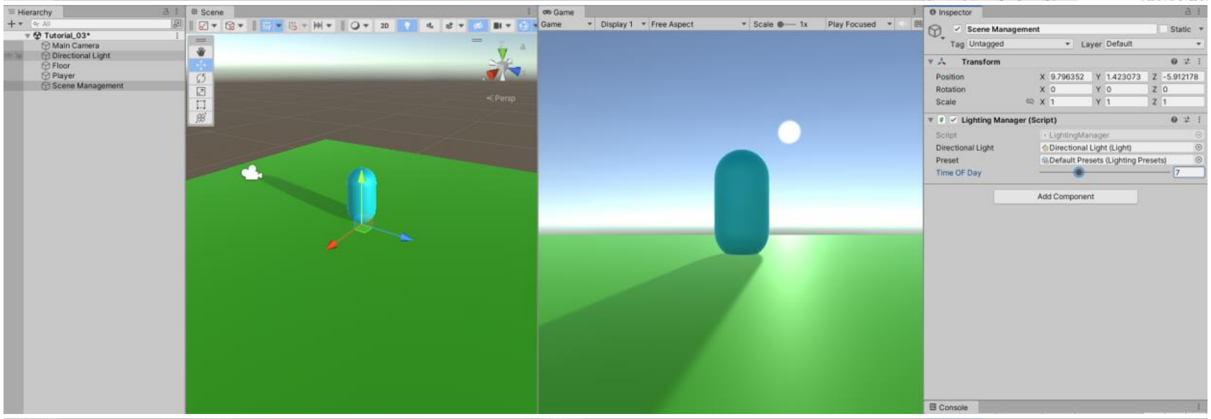
Scene Management | Static
Tag Untagged | Layer Default

Transform

Position	X	9.796352	Y	1.423073	Z	-5.912178
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Lighting Manager (Script)

Script	LightingManager
Directional Light	Directional Light (Light)
Preset	Default Presets (Lighting Presets)
Time OF Day	5



Hierarchy
- Tutorial_03*
  - Main Camera
  - Directional Light
  - Floor
  - Player
  - Scene Management

Inspector

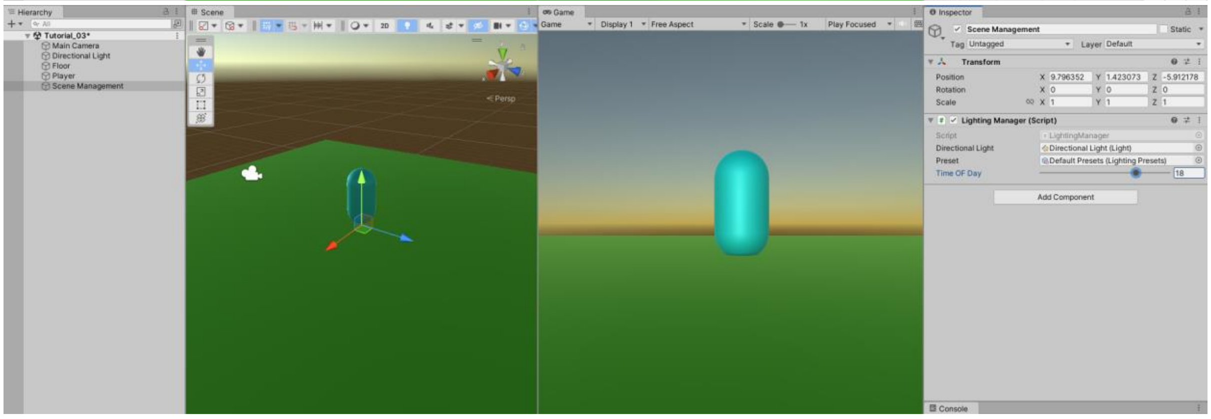
Scene Management | Static
Tag Untagged | Layer Default

Transform

Position	X	9.796352	Y	1.423073	Z	-5.912178
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Lighting Manager (Script)

Script	LightingManager
Directional Light	Directional Light (Light)
Preset	Default Presets (Lighting Presets)
Time OF Day	7



Hierarchy
- Tutorial_03*
  - Main Camera
  - Directional Light
  - Floor
  - Player
  - Scene Management

Inspector

Scene Management | Static
Tag Untagged | Layer Default

Transform

Position	X	9.796352	Y	1.423073	Z	-5.912178
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Lighting Manager (Script)

Script	LightingManager
Directional Light	Directional Light (Light)
Preset	Default Presets (Lighting Presets)
Time OF Day	18