SKYBOX BLENDER Documentation

For support or questions:

visit the Pathiral Discord server here

or email me directly: pathiralgames@gmail.com

Check out more of Pathiral tools & packages here

Skybox Blender is a tool that helps you blend and transition between skyboxes linearly. There is no amount limit to how many skyboxes you can transition to. The entire shader and bunch of other stuff are abstracted through a simple custom inspector tool that'll make your life easier and your job quicker.

GETTING STARTED: (Tutorial Video)

1. Create an empty game object and add the script ***SkyboxBlender*** to it.
2. Set the value of *SkyboxMaterials* to 1.
3. Drag and drop a custom skybox material to the empty value. This will be the skybox you're going to transition/blend into.
4. Drag and drop another custom skybox material to your scene. So your scene has a skybox. **Your scene must have a skybox material added! It can't be empty.**
5. Now make a new script, let's call it **pressSpacebar** but you can call it anything you like. This new script will activate the blending to the new skybox when we press spacebar.
6. Inside this new script write/paste this:

```
using UnityEngine;

public class pressSpacebar : MonoBehaviour
{
    public SkyboxBlender skyboxScript;


    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Space)) {
            skyboxScript.Blend();
        }
    }
}
```

7. Now get back to the editor and add this script to the same empty game object which has the *SkyboxBlender script.*
8. Drag and drop the SkyboxBlender script component to the empty value of skyboxScript.
9. Now play the game and press spacebar.
10. Your skyboxes will blend and transition from one to another.

PROPERTIES & APIs:

***Blend(bool singlePass=false, bool rotate=true)* ->** This is a public method for the skybox blending and it takes two bool arguments which have default values. If you call Blend(); you will loop through all the materials in the list until you reach the end of the list. If loop is set, it will reset to the first skybox and start all over.

However, if you call Blend(true); you will call for a single blend. Which means it will only blend to the next skybox. If you call it again, it will again blend to the next skybox and so on. Until it reaches the end of the materials list in which it will reset to the first skybox.

The second parameter is whether you want to rotate the skybox along the blending or not. By default it's set to true so will rotate when blending is called.

EXAMPLE USE CASES:

Blend(false, true) = will blend through all the materials in the list one by one while rotating. This is the default and it's the same as calling Blend();

Blend(true, false) = will blend only to the next skybox in the list with no rotation.

Blend(true, true) = will blend to the next skybox only while rotating.

Blend(int skyboxMaterialIndex, bool rotate = true) → The very same function as the one above can be used by passing an index of the material you want to transition to. The index passed should be that of the material in the *skyboxMaterials.* The second argument is again for the rotation with a default value of true.

***Stop(bool stoprotate=true)* ->** This is a public method to stop the skybox blending at anytime. It takes an argument of type bool that is automatically set to true. The passed argument is whether you want to stop the rotation as well along side the blending. By default it'll stop the rotation AND the blending.

***Cancel()* ->** Will cancel the current blend and reset the skybox to the previous one.

***Resume(bool resumeRotation=true)* ->** If you called Stop() and want to resume the blending you can either call this function or call the same Blend() method you called before. Both will resume the blending you had before. It takes an argument which is set to true by default. The passed argument is whether you want to resume the rotation as well or not.

IsBlending() -> Returns a bool whether there is an active blending process or not.

Rotate() -> Rotate the skybox directly without blending.

StopRotation() -> Stops the rotation of the skybox.

CurrentIndex -> (**int**) get the current index of skybox material. *Take note:* The index of the skybox changes only after the blending is 100% complete.

P.S: FOR YOUR CURIOSITY AND KNOWLEDGE, BOTH THE MATERIAL AND SHADER USED IN THIS SYSTEM ARE LOCATED INSIDE SKYBOX BLENDER > RESOURCES > MATERIAL & SHADER > …<HERE> YOU CAN MANUALLY USE IT FOR YOUR NEEDS AND FOR MORE CONTROL OVER THE BLEND VALUE