master 1 branch 0 tags Go to file Add file Code

YuSiangLai Update README.md 4ac66ef on Jan 29, 2021 11 commits

Core	support render scale setting	13 months ago
README.assets	update README and rename shader path	13 months ago
TestBed	update README and rename shader path	13 months ago
VolumeComponent	fix wrong parameter	13 months ago
Core.meta	upload version 0.1	14 months ago
LICENSE.md	add CloudShadow post-processing	13 months ago
LICENSE.md.meta	add CloudShadow post-processing	13 months ago
PostProcessForwardRendererData.ass...	update README and rename shader path	13 months ago
PostProcessForwardRendererData.ass...	update README and rename shader path	13 months ago
PostProcessOrderConfig.asset	update README and rename shader path	13 months ago
PostProcessOrderConfig.asset.meta	upload version 0.1	14 months ago
README.assets.meta	update README and rename shader path	13 months ago
README.md	Update README.md	13 months ago
README.md.meta	upload version 0.1	14 months ago
SleeplessOwl.URPPostProcessing.asm...	update util	14 months ago
SleeplessOwl.URPPostProcessing.asm...	update util	14 months ago
TestBed.meta	add test scene	13 months ago
VolumeComponent.meta	add CloudShadow post-processing	13 months ago
package.json	update README and rename shader path	13 months ago
package.json.meta	update README and rename shader path	13 months ago

README.md

Intro

一個 URP 自訂義後處理框架，類似於HDRP攜展後處理的方式，另包含了一些自己的實作。

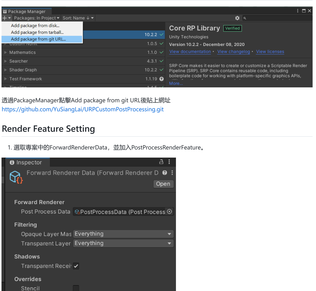
雖然URP目前可以每個後處理自己寫一個RendererFeature，但重數量多時相當麻煩，所以參照HDRP的方式寫了一個URP版本。

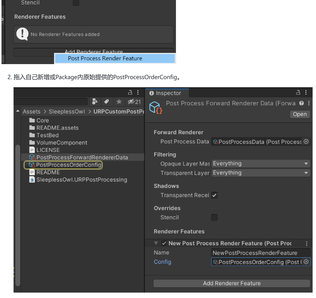
Feature

- 可自己排序後處理執行順序
- 支援新版Unity的 `Volume System`

Content

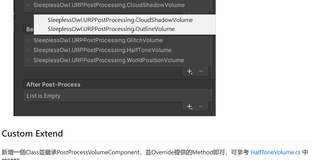
- Outline
- Glitch
- HalfTone
- WorldPosition (need enable DepthTexture in UniversalRenderPipelineAsset)
- CloudShadow (need enable DepthTexture in UniversalRenderPipelineAsset)





How to use

Instal



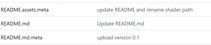
透過PackageManager點擊Add package from git URL後貼上網址
https://github.com/YuSiangLai/URPCustomPostProcessing.git

Render Feature Setting

1. 選取專案中的ForwardRendererData，並加入PostProcessRenderFeature。

2. 拖入自己新增或Package內原始提供的PostProcessOrderConfig。

3. 新增並排序自己需要的後處理功能後，即可在場景中透過 `Volume Component` 使用。



Custom Extend

新增一個Class並繼承PostProcessVolumeComponent，並Override提供的Method即可，可參考 HalfToneVolume.cs 中的註解。

```
using System;
using UnityEngine;
using UnityEngine.Rendering;
namespace SleeplessOwl.URPPostProcessing
{
    [Serializable, VolumeComponentMenu("SleeplessOwl PostProcessing/HalfTone")]
    public sealed class HalfToneVolume : PostProcessVolumeComponent
    {
        public BoolParameter _visibleInSceneview = new BoolParameter(true);
        public FloatParameter _Density = new FloatParameter(0);
        public FloatParameter _Radius = new FloatParameter(0.1f);
        public FloatParameter _SmoothEdge = new FloatParameter(0.2f);
        public FloatParameter _HalfToneFactor = new FloatParameter(0.5f);
        public FloatParameter _SourceFactor = new FloatParameter(0.5f);
        public FloatParameter _Lightness = new FloatParameter(1);
        public ColorParameter _PointColor = new ColorParameter(new Color(0, 0, 0, 1));
        public ColorParameter _ColorFactor = new ColorParameter(new Color(1, 1, 1, 1));

        private Material material;

        //Define this Post-Processing will be executed in which timing in URP pipeline.
        public override InjectionPoint InjectionPoint => InjectionPoint.BeforePostProcess;

        //Default is true.
        public override bool visibleInSceneView => _visibleInSceneview.value;

        //If return false, will skip this Post-Processing.
        //You should ensure the default parameter value will return false in this method, let disable volume
        public override bool IsActive() => _Density.value != 0;

        //Cache parameter ids to avoid do the same thing every update.
        static class Ids
        {
            internal readonly static int _Density = Shader.PropertyToID("_Density");
            internal readonly static int _Radius = Shader.PropertyToID("_Radius");
            internal readonly static int _SmoothEdge = Shader.PropertyToID("_SmoothEdge");
            internal readonly static int _HalfToneFactor = Shader.PropertyToID("_HalfToneFactor");
            internal readonly static int _SourceFactor = Shader.PropertyToID("_SourceFactor");
            internal readonly static int _Lightness = Shader.PropertyToID("_Lightness");
            internal readonly static int _Color01 = Shader.PropertyToID("_Color01");
            internal readonly static int _ColorFactor = Shader.PropertyToID("_ColorFactor");
        }

        //Create shader material.
        //You should put shader in the Resources folder, ensure it will be included in Asset Bundle, or add i
        public override void Initialize()
        {
            material = CoreUtils.CreateEngineMaterial("SleeplessOwl/Post-Processing/HalfTone");
        }

        public override void Render(CommandBuffer cb, Camera camera, RenderTargetIdentifier source, RenderTar
        {
            //Update parameter
            material.SetFloat(Ids._Density, _Density.value);
            material.SetFloat(Ids._Radius, _Radius.value);
            material.SetFloat(Ids._SmoothEdge, _SmoothEdge.value);
            material.SetFloat(Ids._HalfToneFactor, _HalfToneFactor.value);
            material.SetFloat(Ids._SourceFactor, _SourceFactor.value);
            material.SetFloat(Ids._Lightness, _Lightness.value);
            material.SetColor(Ids._Color01, _PointColor.value);
            material.SetColor(Ids._ColorFactor, _ColorFactor.value);

            //Set RenderTexture for shader use.
            cb.SetPostProcessSourceTexture(source);

            //Set render target and draw.
            cb.DrawFullScreenTriangle(material, destination);
        }

        //Do something before Destroy(), if you need.
        protected override void Cleanup()
        {

        }
    }
}
```

Reference

- https://github.com/keijiro/Kino
- https://github.com/Unity-Technologies/Graphics/tree/master/com.unity.testing.hdrp