using System;

using UnityEngine;

namespace UnityStandardAssets.ImageEffects

{

[ExecuteInEditMode]

[RequireComponent (typeof(Camera))]

[AddComponentMenu ("Image Effects/Other/Screen Overlay")]

public class ScreenOverlay : PostEffectsBase

{

public enum OverlayBlendMode

{

Additive = 0,

ScreenBlend = 1,

Multiply = 2,

Overlay = 3,

AlphaBlend = 4,

}

public OverlayBlendMode blendMode = OverlayBlendMode.Overlay;

public float intensity = 1.0f;

public Texture2D texture = null;

public Shader overlayShader = null;

private Material overlayMaterial = null;

public override bool CheckResources ()

{

CheckSupport (false);

overlayMaterial = CheckShaderAndCreateMaterial (overlayShader, overlayMaterial);

if (!isSupported)

ReportAutoDisable ();

return isSupported;

}

void OnRenderImage (RenderTexture source, RenderTexture destination)

{

if (CheckResources() == false)

{

Graphics.Blit (source, destination);

return;

}

Vector4 UV\_Transform = new Vector4(1, 0, 0, 1);

#if UNITY\_WP8

// WP8 has no OS support for rotating screen with device orientation,

// so we do those transformations ourselves.

if (Screen.orientation == ScreenOrientation.LandscapeLeft) {

UV\_Transform = new Vector4(0, -1, 1, 0);

}

if (Screen.orientation == ScreenOrientation.LandscapeRight) {

UV\_Transform = new Vector4(0, 1, -1, 0);

}

if (Screen.orientation == ScreenOrientation.PortraitUpsideDown) {

UV\_Transform = new Vector4(-1, 0, 0, -1);

}

#endif

overlayMaterial.SetVector("\_UV\_Transform", UV\_Transform);

overlayMaterial.SetFloat ("\_Intensity", intensity);

overlayMaterial.SetTexture ("\_Overlay", texture);

Graphics.Blit (source, destination, overlayMaterial, (int) blendMode);

}

}

}