using System;

using UnityEditor;

using UnityEngine;

namespace UnityStandardAssets.ImageEffects

{

[CustomEditor (typeof(EdgeDetection))]

class EdgeDetectionEditor : Editor

{

SerializedObject serObj;

SerializedProperty mode;

SerializedProperty sensitivityDepth;

SerializedProperty sensitivityNormals;

SerializedProperty lumThreshhold;

SerializedProperty edgesOnly;

SerializedProperty edgesOnlyBgColor;

SerializedProperty edgeExp;

SerializedProperty sampleDist;

void OnEnable () {

serObj = new SerializedObject (target);

mode = serObj.FindProperty("mode");

sensitivityDepth = serObj.FindProperty("sensitivityDepth");

sensitivityNormals = serObj.FindProperty("sensitivityNormals");

lumThreshhold = serObj.FindProperty("lumThreshhold");

edgesOnly = serObj.FindProperty("edgesOnly");

edgesOnlyBgColor = serObj.FindProperty("edgesOnlyBgColor");

edgeExp = serObj.FindProperty("edgeExp");

sampleDist = serObj.FindProperty("sampleDist");

}

public override void OnInspectorGUI () {

serObj.Update ();

GUILayout.Label("Detects spatial differences and converts into black outlines", EditorStyles.miniBoldLabel);

EditorGUILayout.PropertyField (mode, new GUIContent("Mode"));

if (mode.intValue < 2) {

EditorGUILayout.PropertyField (sensitivityDepth, new GUIContent(" Depth Sensitivity"));

EditorGUILayout.PropertyField (sensitivityNormals, new GUIContent(" Normals Sensitivity"));

}

else if (mode.intValue < 4) {

EditorGUILayout.PropertyField (edgeExp, new GUIContent(" Edge Exponent"));

}

else {

// lum based mode

EditorGUILayout.PropertyField (lumThreshhold, new GUIContent(" Luminance Threshold"));

}

EditorGUILayout.PropertyField (sampleDist, new GUIContent(" Sample Distance"));

EditorGUILayout.Separator ();

GUILayout.Label ("Background Options");

edgesOnly.floatValue = EditorGUILayout.Slider (" Edges only", edgesOnly.floatValue, 0.0f, 1.0f);

EditorGUILayout.PropertyField (edgesOnlyBgColor, new GUIContent (" Color"));

serObj.ApplyModifiedProperties();

}

}

}