using System;

using UnityEngine;

using Object = UnityEngine.Object;

// same as Triangles but creates quads instead which generally

// saves fillrate at the expense for more triangles to issue

namespace UnityStandardAssets.ImageEffects

{

class Quads

{

static Mesh[] meshes;

static int currentQuads = 0;

static bool HasMeshes ()

{

if (meshes == null)

return false;

foreach (Mesh m in meshes)

if (null == m)

return false;

return true;

}

public static void Cleanup ()

{

if (meshes == null)

return;

for (int i = 0; i < meshes.Length; i++)

{

if (null != meshes[i])

{

Object.DestroyImmediate (meshes[i]);

meshes[i] = null;

}

}

meshes = null;

}

public static Mesh[] GetMeshes ( int totalWidth, int totalHeight)

{

if (HasMeshes () && (currentQuads == (totalWidth \* totalHeight))) {

return meshes;

}

int maxQuads = 65000 / 6;

int totalQuads = totalWidth \* totalHeight;

currentQuads = totalQuads;

int meshCount = Mathf.CeilToInt ((1.0f \* totalQuads) / (1.0f \* maxQuads));

meshes = new Mesh [meshCount];

int i = 0;

int index = 0;

for (i = 0; i < totalQuads; i += maxQuads)

{

int quads = Mathf.FloorToInt (Mathf.Clamp ((totalQuads-i), 0, maxQuads));

meshes[index] = GetMesh (quads, i, totalWidth, totalHeight);

index++;

}

return meshes;

}

static Mesh GetMesh (int triCount, int triOffset, int totalWidth, int totalHeight)

{

var mesh = new Mesh ();

mesh.hideFlags = HideFlags.DontSave;

var verts = new Vector3[triCount \* 4];

var uvs = new Vector2[triCount \* 4];

var uvs2 = new Vector2[triCount \* 4];

var tris = new int[triCount \* 6];

for (int i = 0; i < triCount; i++)

{

int i4 = i \* 4;

int i6 = i \* 6;

int vertexWithOffset = triOffset + i;

float x = Mathf.Floor (vertexWithOffset % totalWidth) / totalWidth;

float y = Mathf.Floor (vertexWithOffset / totalWidth) / totalHeight;

Vector3 position = new Vector3 (x \* 2 - 1, y \* 2 - 1, 1.0f);

verts[i4 + 0] = position;

verts[i4 + 1] = position;

verts[i4 + 2] = position;

verts[i4 + 3] = position;

uvs[i4 + 0] = new Vector2 (0.0f, 0.0f);

uvs[i4 + 1] = new Vector2 (1.0f, 0.0f);

uvs[i4 + 2] = new Vector2 (0.0f, 1.0f);

uvs[i4 + 3] = new Vector2 (1.0f, 1.0f);

uvs2[i4 + 0] = new Vector2 (x, y);

uvs2[i4 + 1] = new Vector2 (x, y);

uvs2[i4 + 2] = new Vector2 (x, y);

uvs2[i4 + 3] = new Vector2 (x, y);

tris[i6 + 0] = i4 + 0;

tris[i6 + 1] = i4 + 1;

tris[i6 + 2] = i4 + 2;

tris[i6 + 3] = i4 + 1;

tris[i6 + 4] = i4 + 2;

tris[i6 + 5] = i4 + 3;

}

mesh.vertices = verts;

mesh.triangles = tris;

mesh.uv = uvs;

mesh.uv2 = uvs2;

return mesh;

}

}

}