Native Texture Renderer – Android

Pre Requisites:

- Native Texture Renderer – Android is a Library for Rendering up to 8k Textures asynchronously without blocking the Main Thread. It is heavily Multithreaded and Every Upload Request to the GPU is Queued for maximum memory efficiency.

- This Plugin works with Both OpenGL ES 2 & 3 Devices. Minimum SDK Requirements depends on them.

- An Example Scene is provided under "$ProjectPath/Assets/NativeTextureRenderer/Scenes/" Which shows all the working of the Library.

- The Plugins Folder Present in "$ProjectPath/Assets/NativeTextureRenderer/Plugins" Must be copied to "$ProjectPath/Assets/Plugins/"

- For The Demo Scene to work properly you need to Copy the Textures from /Assets/NativeTextureRenderer/Textures/*.jpg files into the "PersistentDataPath" i.e, /YourSDCard/Android/Data/com.stuxnetgs.androidnativetexturerenderer/Files/"Here".

- If you have any problems or queries you can email me at : stuxnetgs@gmail.com along with your order number. Will respond ASAP.

MANUAL:

- Little setup is required i.e., Copying the contents in the Plugins Directory to the /Assets/Plugins/ Path as described before.
- The Library only works with ARMv7 CPU Architectures.
- There will be no conflicts as the Libraries are Shared Libraries.

DOCUMENTATION:

- There is no need to Include Any Namespace.
- Create an Empty Game Object in your Scene. Name it to what ever you desire. Go to Add Component > Type in “Native Texture Renderer” and add the script to it. That’s all you need to set it up.

For Textures on Local Storage:

- To Render a Texture, you need the Path to the texture any where on the local disk (Make sure your App has necessary permissions to read the storage medium).

For Textures that are to be Downloaded:

- If you need to Render a Texture that you need to Download. Use “UnityWebRequest” and save it to the Disk. Then use that Path to Render the Texture using the library. I will Add an Utility Soon that will do all this Seamlessly. But for now this is the way to do it.

You Only need to Call One Function to Render Your Texture

- First Create your Texture2D. The Textures need to be in ARGB32 Format.
- Eg : Texture2D tex = new Texure2D(2048,2048, TexureFormat.ARGB32, false);

Static void NativeTextureRenderer.RenderTexture(string PathToTheTexture, Intptr TexturePointer, int Width, int Height);

To get the TexturePointer simply call “Texture2D.GetNativeTexturePtr()” On your created Texture2D.

It is Highly Recommended that you Initialize or Create your Textures in the Start() as creation of Textures in Unity is associated with a slight overhead on Mobile Platforms.

EXAMPLE:

```
1.  using System.Collections;
2.  using UnityEngine;
3.
4.  public class RenderExample : MonoBehaviour
5.  {
6.
7.      public string[] FileName;
8.      public Material[] Mat;
9.      public Texture2D[] tex;
10.
11.     IEnumerator Start()
12.     {
13.         tex = new Texture2D[Mat.Length];
14.         for (int i = 0; i < tex.Length; i++) {
15.             tex[i] = new Texture2D (2048, 2048, TextureFormat.ARGB32, false);
16.         }
17.         yield return true;
18.     }
19.
20.     public void RenderFunc ()
21.     {
22.         string datapath = Application.persistentDataPath;
23.         for (int i = 0; i < Mat.Length; i++) {
24.             string TexturePath = datapath + "/" + FileName [i];
25.             NativeTextureRenderer.RenderTexture (TexturePath,tex[i].GetNativeTextur
    ePtr(),tex[i].width,tex[i].height);
26.             Mat [i].mainTexture = tex[i];
27.         }
28.     }
29. }
```

Function RenderFunc() is Called on a Button Click in the UI. This Renders 4 2k Textures onto 15 cubes. You can Find the Video Demo Here :