

Modular Bricks

- user guide (v 1.0) –

Introduction

Modular Bricks shader pack is a collection of geometry shaders, designed to create **extra geometry** on meshes in order to give them a modular interlocking aspect. The pack comes with two types of bricks: pipes and cubes.



(pipes version of shader on the left side and cubes on the right side)

There is a wide range of shader variant available for both cubes and pipes mode:

1. Diffuse Color
2. Diffuse Texture
3. Diffuse Color Transparent
4. Diffuse Texture Transparent
5. PBL
6. PBL with Emission
7. Vertex Lit
8. Vertex Lit Cull (culling distance for the extra geometry)

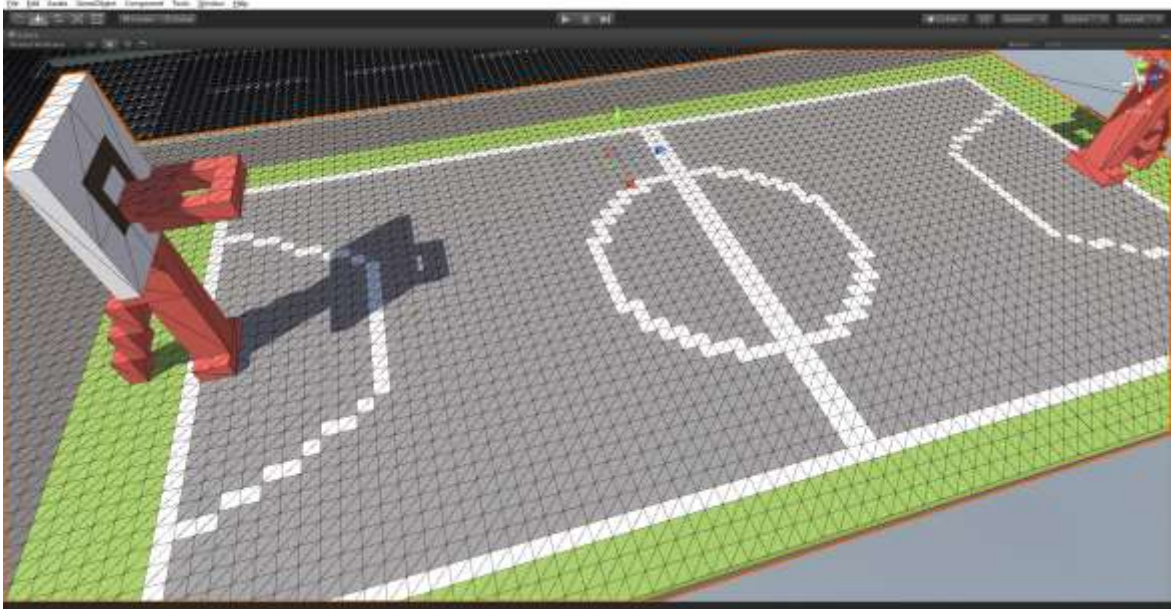
Modular Bricks is compatible with any DirectX11 and OpenGL ES 3.2 or OpenGL ES 3.1 with AEP. On mobile I tested the shader on a Samsung Galaxy S7 with Adreno 530 GPU.

Important Notes

1. The extra geometry drawn by the shader (cubes or pipes) **does not** have custom UV mapping, it inherits the base UV mapping layout from the quad it is drawn onto.
2. Due to the inexistence of a custom UV map there is no compatibility with lightmaps. There will be a shader version in the future that will provide Lightmapping support for base geometry.

Art Creation Guidelines

1. Keep in mind that only the faces with the normal pointing upwards are used for creating the extra geometry.
2. It is recommended to use voxel art as often as you can.
3. You need 2 triangles or a quad for every cube, pipe you want to create. Therefore, in the case of a really large surface, you have to divide it into multiple quads. This approach will make your scene's triangle count to go higher and may increase rendering times. It is recommended to use LODs.



(the *Basketball Court* model provides a quad for every place where the shader will draw the pipes)

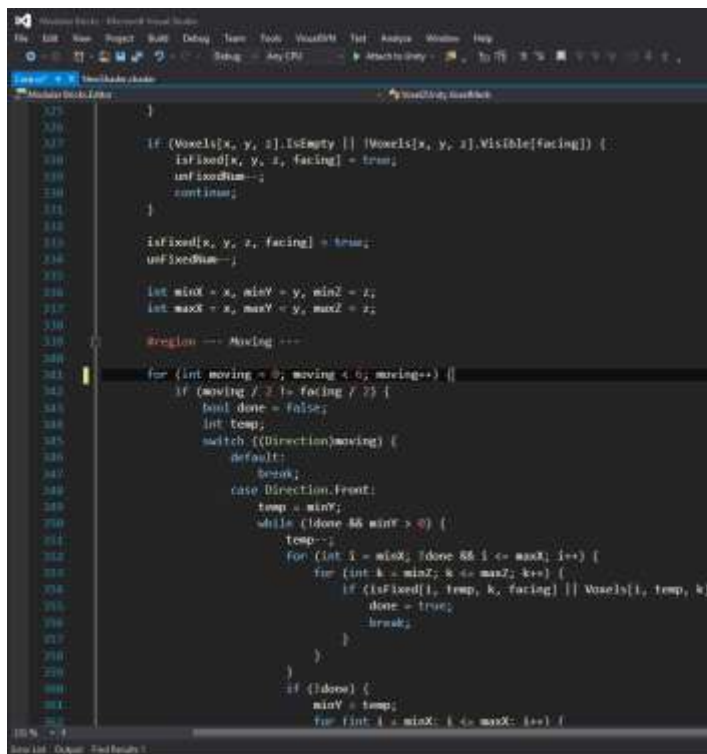
4. A highly-recommended workflow is to create the geometry in [MagicaVoxel](#) and then use [MagicaVoxel Qubicle To Unity](#) plugin in order to import it in Unity (also check compatibility section).

Other Plugins Compatibility

MagicaVoxel Qubicle To Unity ([link](#))

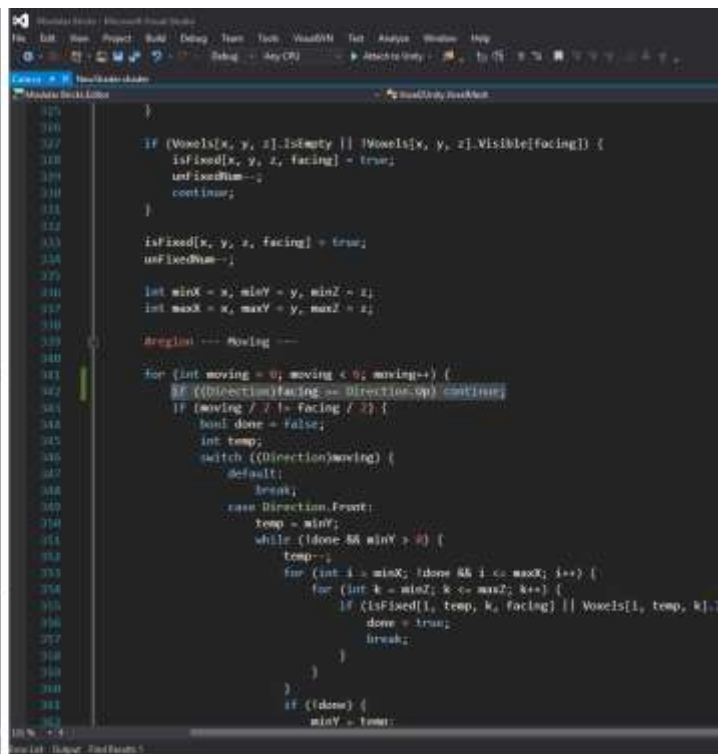
In order to make this plugin compatible with Modular Bricks shader pack you have to avoid the top faces from being merged together.

1. Import MagicaVoxel Qubicle To Unity
2. Go to *VoxelToUnity > Editor > Open Core.cs*
3. Go to the end of line 341
4. Press Enter to on a new line and paste:
`if ((Direction)facing == Direction.Up) continue;`



```
325 }
326
327 if (Voxels[x, y, z].IsEmpty || !Voxels[x, y, z].Visible[facing]) {
328     isFixed[x, y, z, facing] = true;
329     unfixedDim--;
330     continue;
331 }
332
333 isFixed[x, y, z, facing] = true;
334 unfixedDim--;
335
336 int minX = x, minY = y, minZ = z;
337 int maxX = x, maxY = y, maxZ = z;
338
339 #region --- Moving ---
340
341 for (int moving = 0; moving < 6; moving++) {
342     if (moving / 2 != facing / 2) {
343         bool done = false;
344         int temp;
345         switch ((Direction)moving) {
346             default:
347                 break;
348             case Direction.Front:
349                 temp = minY;
350                 while ((done && minY > 0)) {
351                     temp--;
352                     for (int i = minX; !done && i <= maxZ; i++) {
353                         for (int k = minZ; k <= maxZ; k++) {
354                             if (!isFixed[i, temp, k, facing] || Voxels[i, temp, k].
355                                 done = true;
356                                 break;
357                         }
358                     }
359                 }
360                 if (!done) {
361                     minY = temp;
362                     for (int i = minX; i <= maxZ; i++) {
```

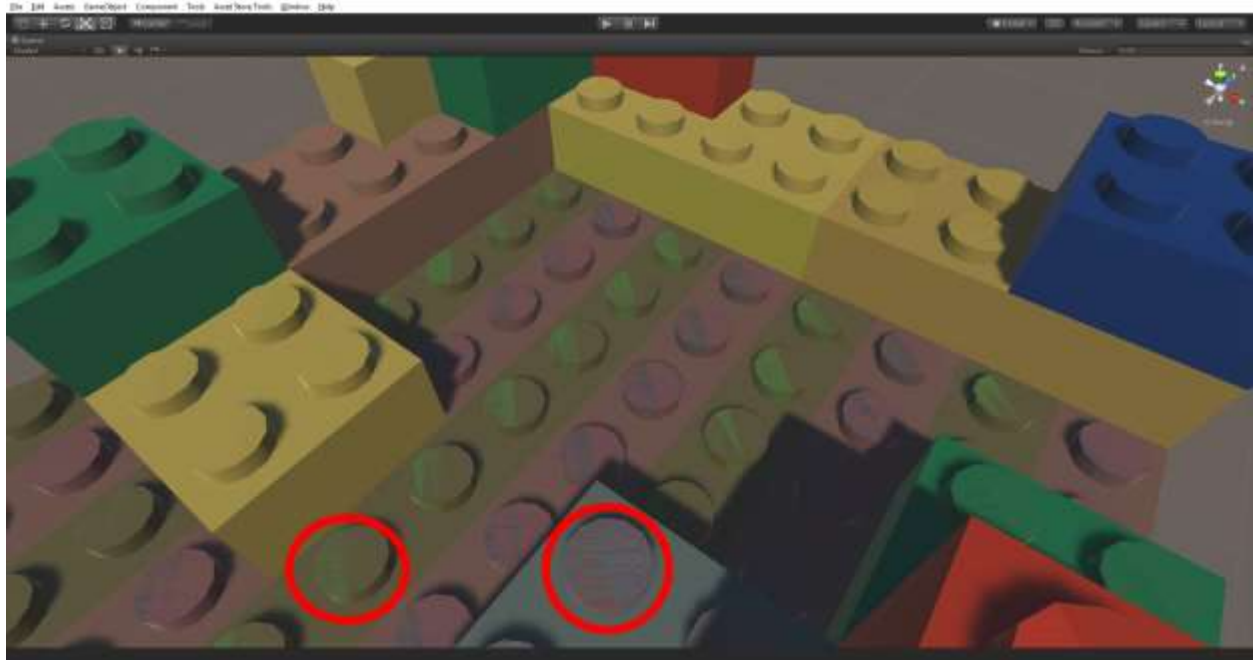
Original



```
325 }
326
327 if (Voxels[x, y, z].IsEmpty || !Voxels[x, y, z].Visible[facing]) {
328     isFixed[x, y, z, facing] = true;
329     unfixedDim--;
330     continue;
331 }
332
333 isFixed[x, y, z, facing] = true;
334 unfixedDim--;
335
336 int minX = x, minY = y, minZ = z;
337 int maxX = x, maxY = y, maxZ = z;
338
339 #region --- Moving ---
340
341 for (int moving = 0; moving < 6; moving++) {
342     if ((Direction)facing == Direction.Up) continue;
343     if (moving / 2 != facing / 2) {
344         bool done = false;
345         int temp;
346         switch ((Direction)moving) {
347             default:
348                 break;
349             case Direction.Front:
350                 temp = minY;
351                 while ((done && minY > 0)) {
352                     temp--;
353                     for (int i = minX; !done && i <= maxZ; i++) {
354                         for (int k = minZ; k <= maxZ; k++) {
355                             if (!isFixed[i, temp, k, facing] || Voxels[i, temp, k].
356                                 done = true;
357                                 break;
358                         }
359                     }
360                 }
361                 if (!done) {
362                     minY = temp;
363                     for (int i = minX; i <= maxZ; i++) {
```

New Code

In case you have problems with the texture as in the following image, make sure to set the texture compression to None and the filter mode to Point.



(you can observe some artifacts on top of the pipes, most likely caused by compression)