Realtime Procedural Brick & Tile Shader

Description:

-Create brick walls and tiled surfaces dynamically, in realtime - no precomputed assets required.
-Texturing is optional (various ways to texture).
-Single pass render does NOT require geometry shader or tesselation.
-Perspective applied to bricks to emulate solid 3D (edges appear raised appropriately).
-Looks good without anti-aliasing (procedural calculations account for this).
-Configurable edge slopes and shapes (flat or rounded).
-Position and sizes of bricks and mortar fully configurable.
-Shader calculates normals for brick shapes so lighting works correctly on edges etc.
-Highly optimized shader code (of course).
-Configurable LOD to enable smooth fade-outs over camera distances.

Installation note:

Move the contents of the Editor folder in BrynderiTech to the Editor folder under Assets. If no Editor folder exists here, move the BrynderiTech Editor folder to be a child of the Assets folder.

Supplied are four shader variants:

ProceduralBrick_Opaque	= Opaque, diffuse lighting
ProceduralBrick_OpaqueSpec	= Opaque, specular lighting
ProceduralBrick_Transparent	= Transparent, diffuse lighting
ProceduralBrick_TransparentSpec	= Transparent, specular lighting

And a custom editor:

ProceduralBrickInspector.cs

Together with scene assets, etc.

For any technical enquiries email **brynderitech@gmail.com**

Brief:

This is a complex shader and this document will break down its operation into several areas, as well as giving a comprehensive list of controls available from the custom editor supplied. You must use the custom editor supplied, since it computes many internal values used in the shader from the settings you give (which means the shader does less work and is faster).

It makes sense to at least read through this document once, since there are quite a few shader controls and options available, and also look at the supplied scene to see how different effects can be produced.

Throughout the document, when we refer to ‘brick’ we mean ‘brick OR tile’ unless we explicitly say that we ONLY mean one or the other, since the only difference is how you configure the settings in the editor - to look like bricks or look like tiles.

Naturally, the shader has some limitations. It does NOT deal with brick walls using different sizes of brick, but deals with the most common configuration of overlapping rows of bricks. However the offsets , sizes and spacing can be configured in the editor - so you can have non-overlapping rows if

you wish, this is particularly useful for producing walls of square tiles like those found in washrooms, operating theaters and so on.

Note that you can alter the edges of the bricks, so they appear sloped, flat or rounded. Bricks will tend to have flat or maybe sloped edges, whereas washroom tiles will tend to have rounded edges and corners. The choice is yours, the shader was designed to be flexible.

Take a look at the sample scenes we built for showing how to use the shader(s), and experiment with the options.

Here's a couple of screenshots from the sample scenes included.



Image - Walls (both brick & tile) and Floor use the brick shader

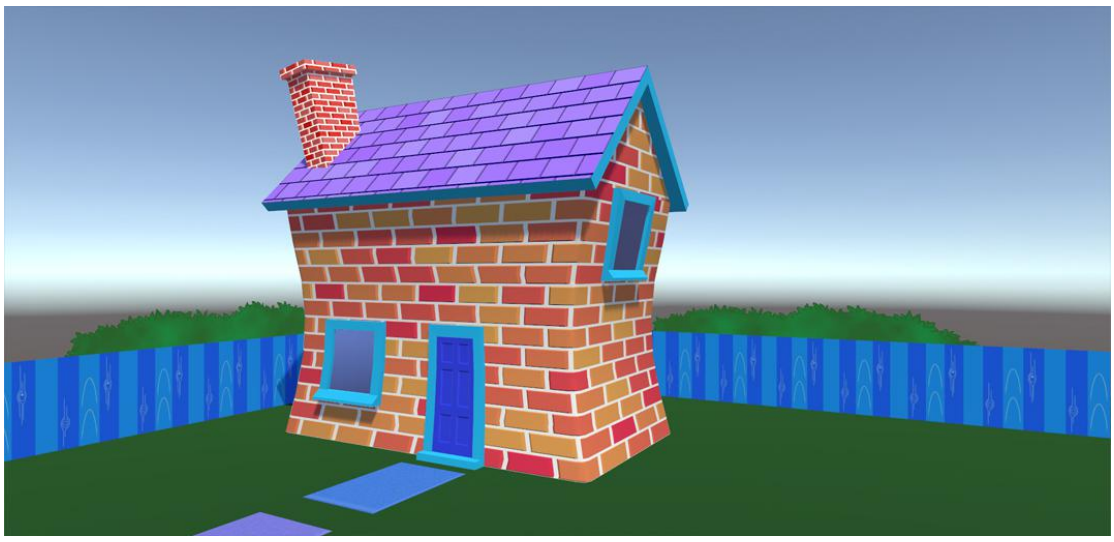
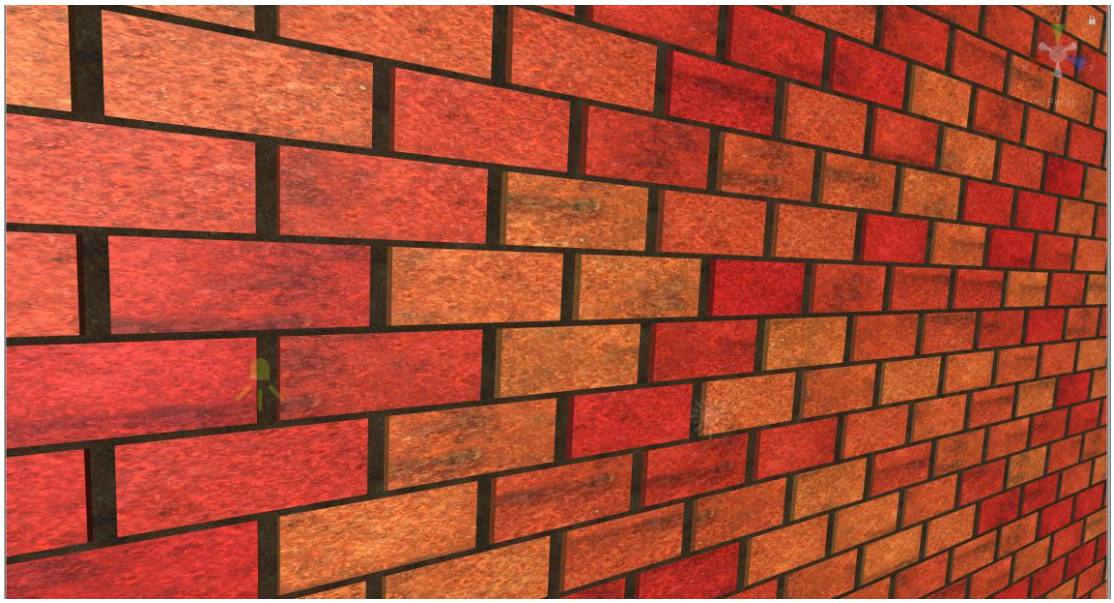


*Image - A cartoon-style house, using the brick-shader **untextured** for walls and roof.*

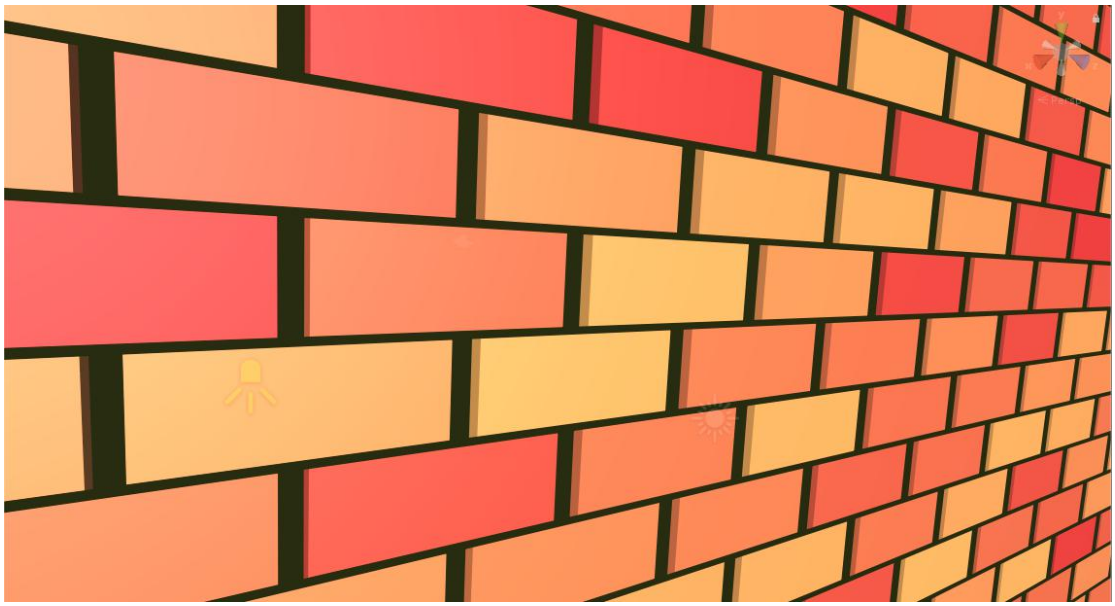
Emulating solidity (3D) bricks on a 2D surface:

It is very important to note here that we do NOT use tesselation, geometry shaders or additional meshes when we render a surface. The illusion of solid bricks is created entirely within the shader, and is calculated according to your eye position and the surface.

For example:



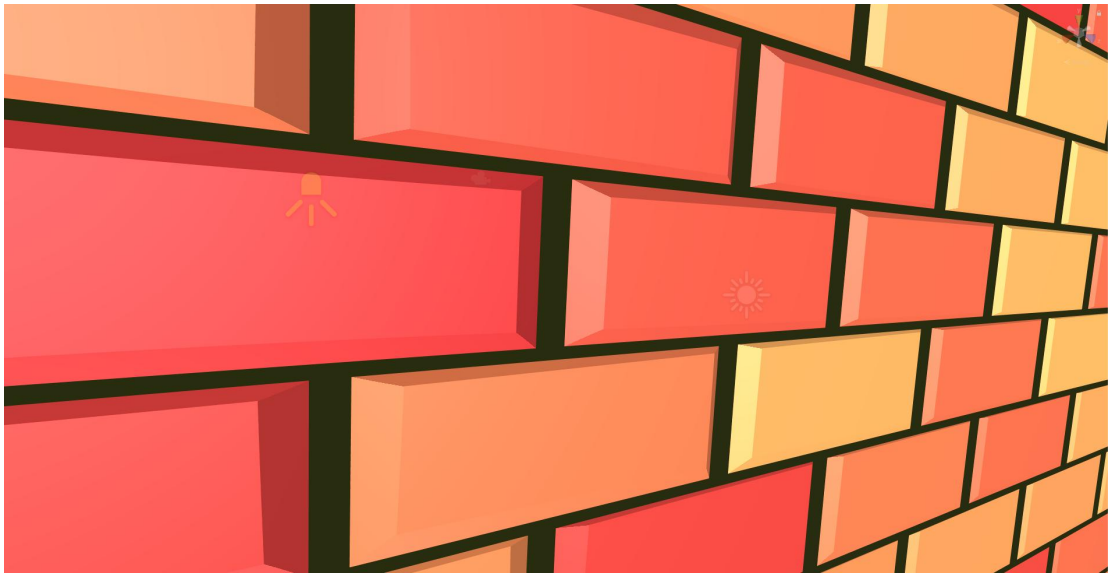
(Bricks - fake3D, textured)



(Bricks - fake3D, untextured)

These were rendered on a flat plane of two triangles. The shader has computed the edges to appear solid.

The edges of the bricks are shaped in the Editor Interface '**Brick Raised Edges**'. You have the option to use raised edges or not. Without it just the pattern of bricks is created, saving a few cycles in the shader! Having said this, the actual calculations are fast, optimized, and will happily work on mobiles.



(Brick with 'sloping' edges, Elevation and Edge thickness adjusted to achieve this)

Limitations of edges:

The edges calculated in the shader are very fast. However there are a few simple limitations on this. The edges of bricks and tiles are quite narrow areas compared to the brick surface, and our system works well provided you do not create hugely wide edge areas. If you do this then you will notice some curvature and inaccuracy. The system is never intended for this, edges are created for emulating the appearance of 3D on bricks, and uses highly optimized calculations which work within the range needed for creating the appearance of solid bricks and tiles, with rounded edges or flat edges similar to what you find in the real world. Normally you will be using Thickness values for edges below 0.2, and with rounded edges you may find higher amounts satisfactory.

The Elevation is also adjusted with this, and provides a way of 'flattening and thickening' the edge. For bricks with flat edges, an elevation of '1' or close to '1' is recommended. For using rounded edges (on tiles for example) it is likely that you will decrease elevation maybe towards 0.5. Experiment using the elevation and edge thickness controls to achieve the desired result.

If the 'edge type' is rounded then you can also control the sharpness of the corners with the 'corner rounding' control in the 'Brick Raised Edges', this is useful on tiles, but as an example here are a couple of screen-shots of our brick wall, untextured, to demonstrate the difference between sharp corners and fully rounded corners when using Rounded edges.

The shots here are untextured so that the effect is obvious and not 'confused' by textures and normal maps being applied.



(Brick with ‘rounded’ edges, normally used for tiles - corners full rounded)



(Brick with ‘rounded’ edges, corners with NO rounding, ie. Sharp)

We adjust the colours, edge thickness, elevation etc. to turn the wall into washroom tiles, in the image below - the elevation is 0.473 and edge thicknesses are about 0.1 - This is probably slightly exaggerated to make the edge more obvious, but the choice is yours of course, as well as the lighting!



(Tile with rounded edges and corners)

Setting the size of bricks and mortar:

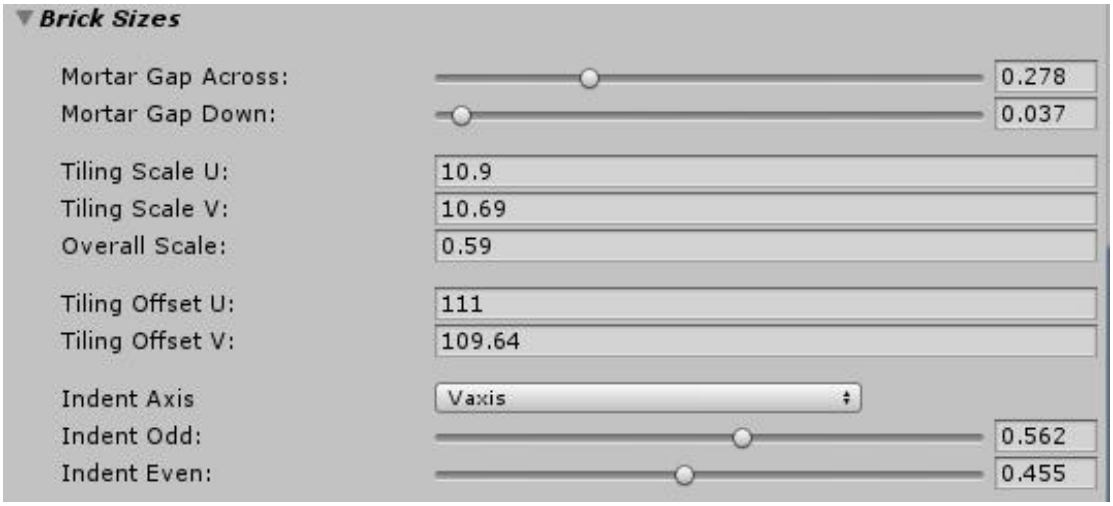
The first thing you need to do will be to set the sizes of bricks, the width of the mortar gap, and also the offset of each row to create the overlap of rows. The '***Brick Sizes***' section under **General Settings** is where you can get started - after setting a range of colors for the bricks (***Coloring bricks***) and a color for the mortar. After all, if the colors are all the same then you'll just get a plain wall of color…

The bricks use the texture UV's of the mesh, and you need to set tiling settings (*Tiling Scale U* and *Tiling Scale V*) to adjust how many bricks you will get in each axis (U and V) - and the offsets (*Tiling Offset U* and *Tiling Offset V*). The bigger the scale numbers, the smaller the bricks and the more of them. Don't be afraid to try big values - since we don't generate any geometry data you can have thousands of bricks on a wall, if you want, and they can all appear solid, it will even work on mobiles (Android 6.0 tested).

To overlap rows of bricks use the 'Indent Axis' to select which direction indenting is to occur. It will either be 'U' or 'V', and this may depend on your texturing and if you are using an automatic texturing method. So just try each to see which is the right one for you.

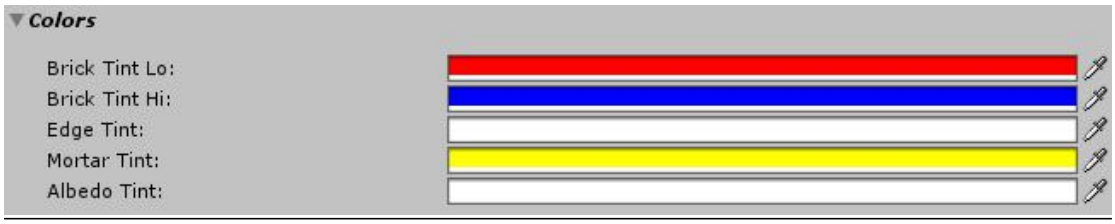
The '*Indent Odd'* and '*Indent Even'* controls allow you to alter the indent on alternate rows of bricks.

Then you decide how much of the 'brick' actually is 'brick' and how much is 'mortar'. This is what the '*Mortar Gap Across*' and '*Mortar Gap Down*' settings do. T hey vary from 0 (minimum)) to 1 (maximum). In reality there is always a small amount of mortar, so it can never be 100% brick - there has to be a tiny mortar gap - its a technical requirement of the shader.



Brick walls usually have each row 'indented' to overlap the one beneath. Washroom tiles typically have no indent. You can select which axis you want to indent (U or V - usually corresponds to across or up/down) and how much of an indent on each row (axis). Being able to indent both odd and even rows helps to match up edges of adjoining walls.

Coloring bricks:



The base color of a brick is randomly chosen from a range of colors you set in the ***Colors*** section in general settings. '*Brick Tint Lo'* and '*Brick Tint Hi'* together set the range of colors to use. If you are texturing the bricks, then the brick texture is modulated with the tint value.

'*Edge Tint*' applies the color you give to the brick edges (it is only available if you have selected **Brick Raised Edges** in the options), after they have been tinted with the Brick Tint. If you are applying texture to these edges, then the texture is tinted accordingly.

Mortar is a constant tint color, from '*Mortar Tint'* setting. This is the color of the mortar. If you are applying a texture to the mortar then the texture is tinted accordingly.

The '*Albedo Tin*t' setting sets the overall light albedo color for the material. Normally it will be set to white, but you can make it feel colder with a pale blue tint, or warmer with a yellow 'Albedo Tint'. This can be useful when you are creating the feel of an early morning or late afternoon in a scene.

To Texture or Not to Texture:

You can turn texturing of both bricks and mortar on or off, as well as normal maps (detail normals).

Why would you want to NOT texture ? - because each brick is tinted randomly with the range of colors you specify (In ***General Settings, Colors)*** you can still produce a nice solid brick wall without using ANY textures - look at the previous images, and this means that you can create assets for low-end platforms (think mobile here!) with no texture overhead.

You can also apply a detail normal map to the brick surface (or mortar surface) WITHOUT having a texture. So its up to you to decide how you want your walls to look.

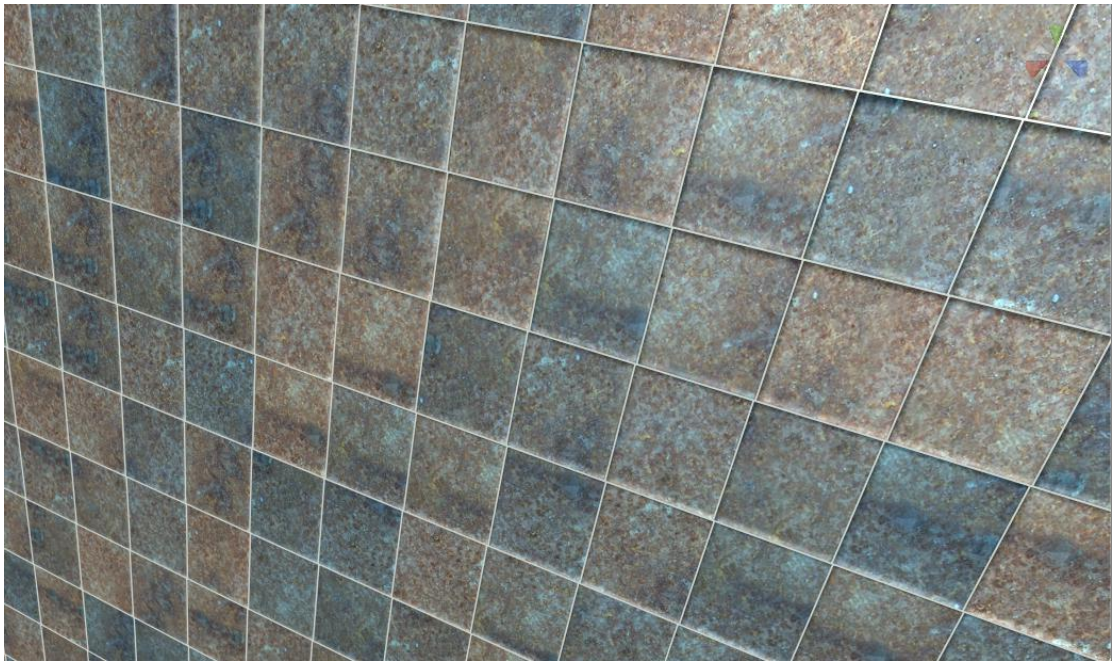
There are three ways you can apply textures -

Mega-texturing
Atlas
Atlas Repeat

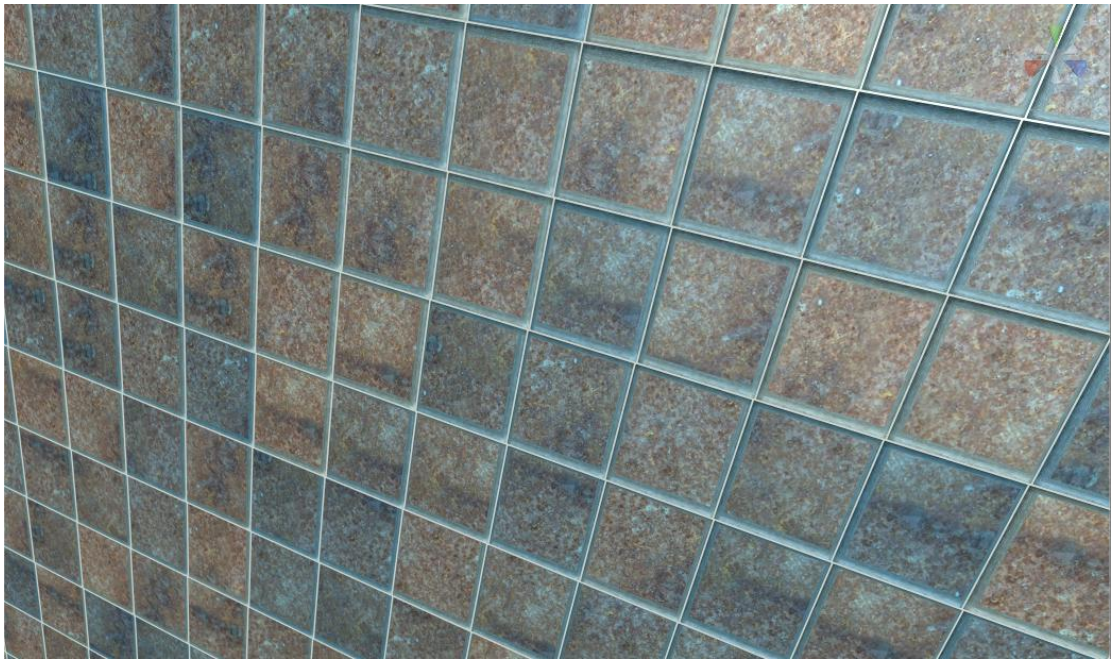
For now, we will just use Mega-texturing to explain how you can either use a single texture to cover the whole brick surface including edges, or use one texture for the flat main region, and another for the edges.

Take a look at the two images below. The first shows the same tiled wall as above (blue tiles) but with a texture applied using Mega-texturing. Note that '*Distinct Edge Texture*' is turned off in the *Brick Raised Edges* controls. The second image shows the same wall, but now we have ticked '*Distinct Edge Texture*' to turn it on - and supplied a different texture for the edges.

The edge texture is rendered using different texture coordinates, so that it will be 'compressed' into the edge.



(Textured using mega-texture, Distinct Edge Texture is OFF)



(Textured using mega-texture, Distinct Edge Texture is ON, and an edge texture supplied)

Mega Texture:

This is just one texture such as brick or stone which tiles in both directions. The shader simply picks a random area of that texture and applies it to each brick. You can randomly flip its orientation in either direction if you wish (the tick-box 'Randomly Flip'). The previous two images demonstrate this. It is VERY important that the texture tiles - or you will get texture seams on some bricks where the texture UV coordinates wrap around.

Important - the texture should be set to 'Repeat' wrap mode in Unity. (In Unity editor, Select the texture and in its Inspector set 'Wrap Mode' to Repeat). If you use a normal map, then the same applies to that texture too (set Wrap Mode to Repeat).

Using '*Distinct Edge Texture':*

Distinct Edge Texture is only available when you are texturing bricks. If you are NOT applying a brick texture then you cannot apply a separate edge texture.

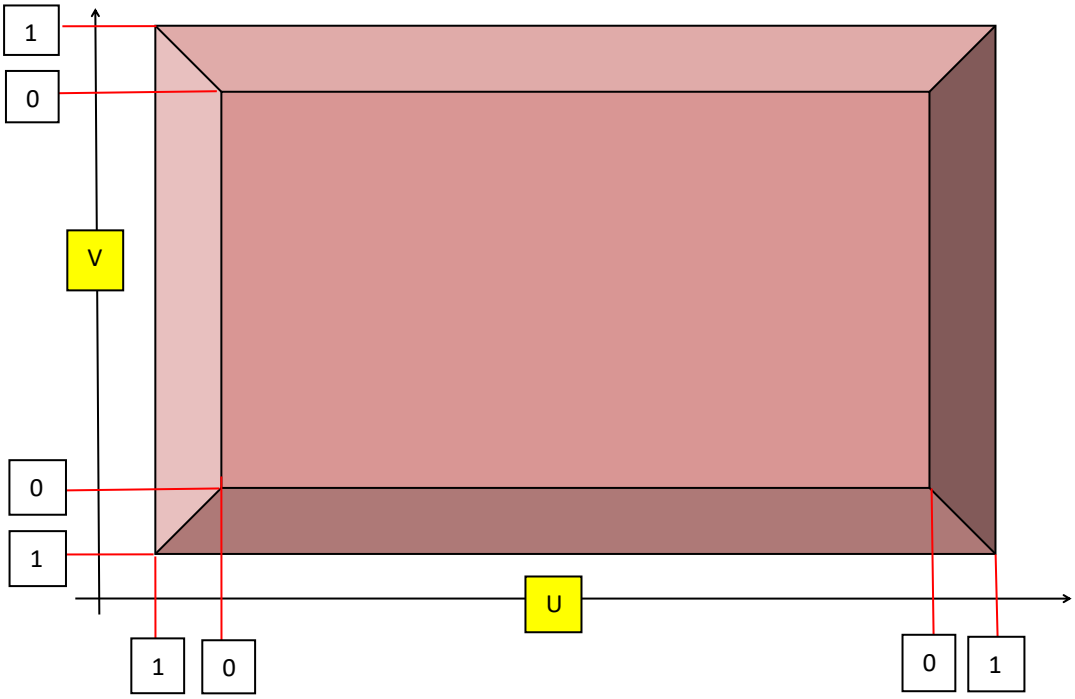
When you select this feature (a tick box in *Brick Raised Edges* section), you are prompted for a texture to apply to the edge regions of bricks.

You may find this useful for creating the appearance of a 'cross-section' on a narrow strip of edge. It can also be really useful when applying textures to tiles. Often a washroom tile will have a pattern texture, but you do not want this applied to the edges, only to the main surface. Using '*Distinct Edge Texture*' will cause the main texture to be 'clamped' to the surface area and then you can apply a different texture to the edge - maybe even a plain white texture.

On bricks that use mega-texture, you may find that you can reapply the SAME texture as the edge-texture and it will look fine.

Remember that edges are only meant to be for narrow strips to simulate solidity- they are not a substitute for a model. And because the edges should be narrow, the shader does NOT support a separate normal map for the *Distinct Edge Texture*, as it would be 'squashed' into a narrow UV zone and simply does not look good.

For reference, the edge texture is rendered in the following way around a brick:

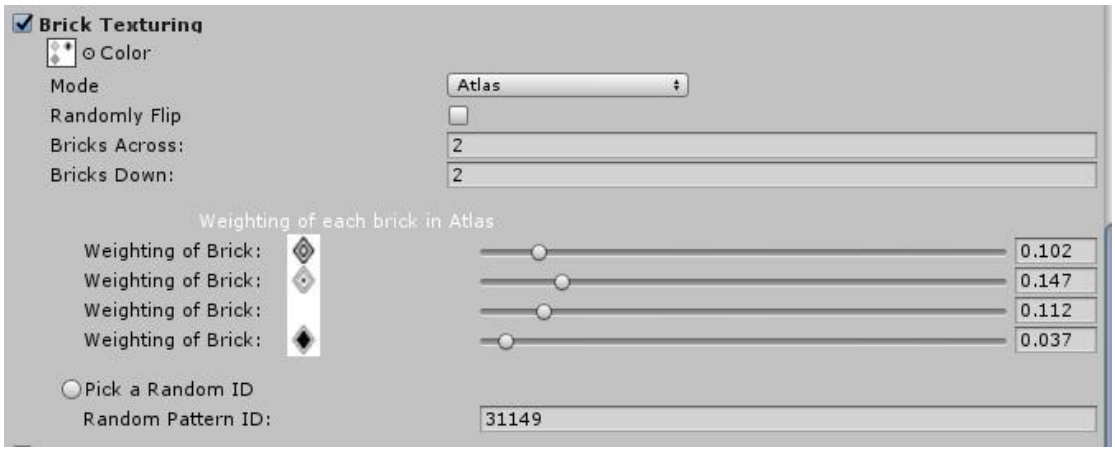


Atlas:

In this texturing mode, the texture you supply for Bricks or tiles consists of a whole set of bricks stored in a single texture. You simply tell the shader how many bricks across and down are stored in the atlas (*Bricks Across* and *Bricks Down* controls) and whether you want the individual bricks randomly flipped and mirrored (*Randomly Flip* tick-box).

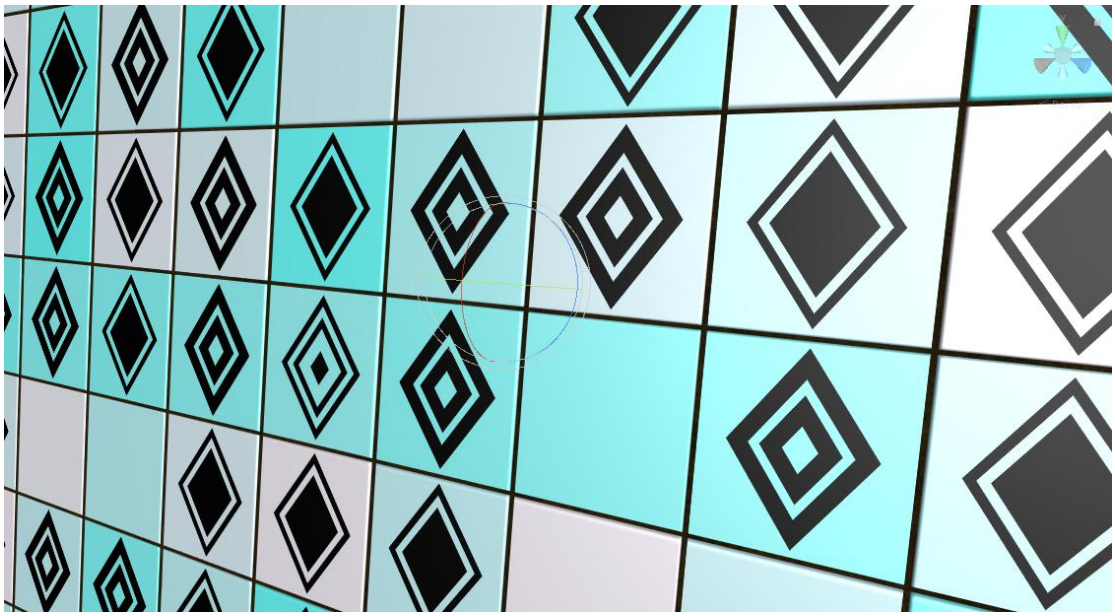
Important - the texture should be set to 'Clamp' mode in Unity. (In Unity editor, Select the texture and in its Inspector set 'Wrap Mode' to Clamp). If you use a normal map, then the same applies to that texture too (set Wrap Mode to Clamp).

You then have a section called 'Weighting of each brick in Atlas' - its in white text to make it show up. This has a slider control for each of the bricks in the atlas, and these control the chance of each brick being selected, ie. Its 'weighting'. The shader selects randomly a brick from the atlas according to its weighting.

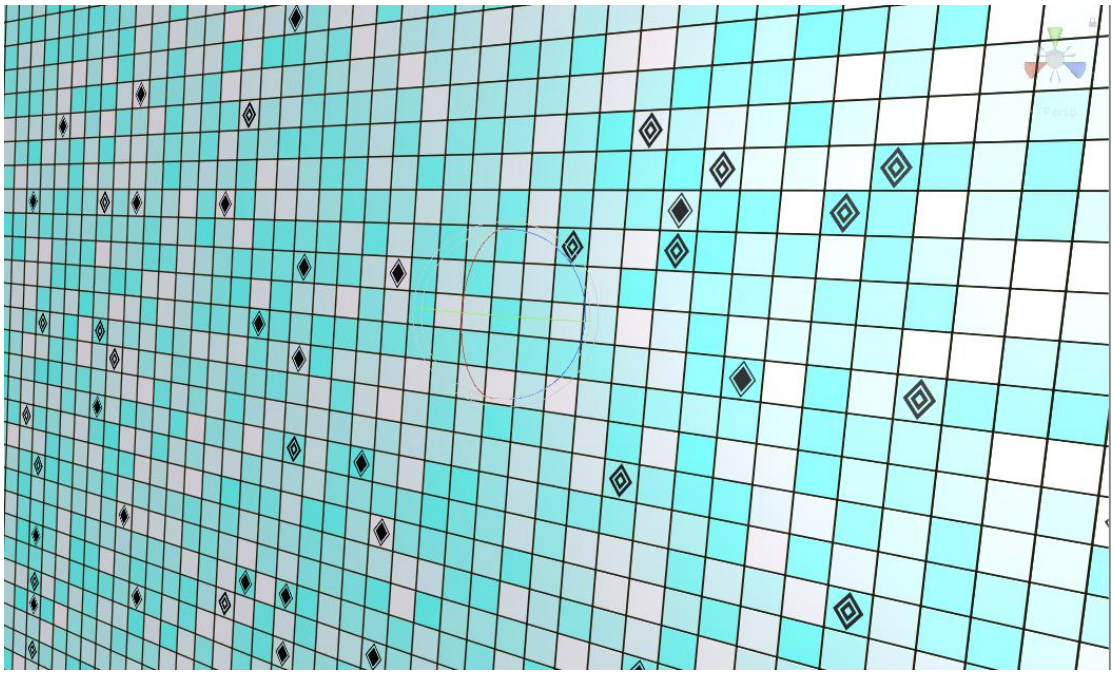


Note that you can change the pattern randomly by selecting a ‘Random Pattern ID’, this value decides the actual pattern produced, change this randomly (press ‘Pick a Random ID’ until you find a pattern you like.

In the above image, taken from the Interface, we are using an atlas of 2 x 2 patterns for washroom tiles. The resulting pattern on a section of tiled wall is shown below:



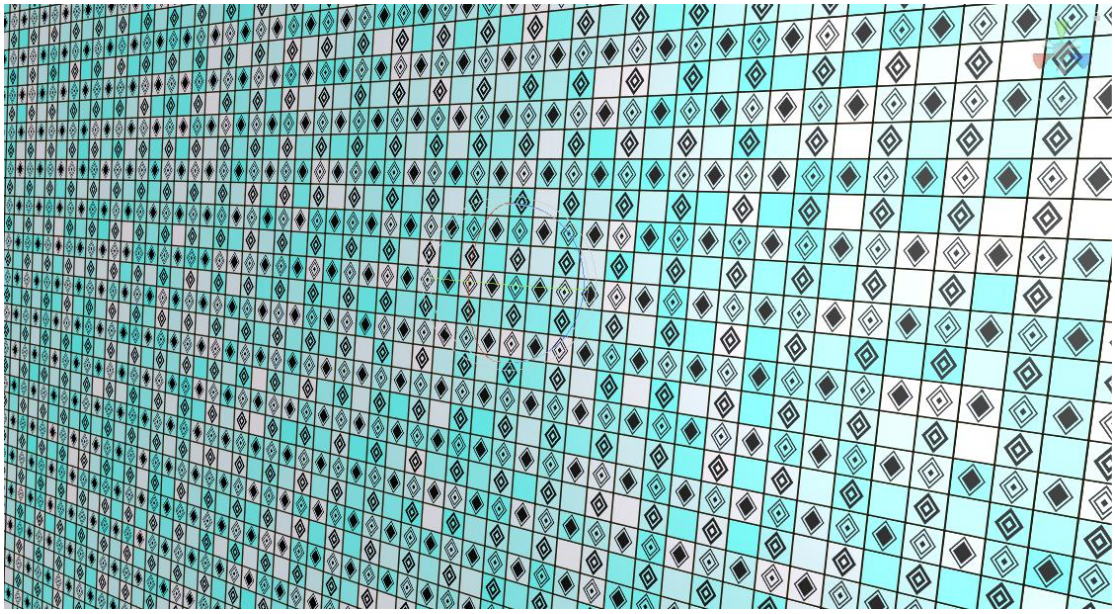
Varying the weightings of each brick (or tile) will allow us to use more of one tile than another. It is common to use a lot of white tiles with only a few patterned ones. Such as the next image, where we have increased the weighting of the one (white) tile and decreased the other weightings.



However, you may decide that you want a whole ‘atlas’ repeated across the tiled surface, not randomly sampled, so the shader supports this with ‘*Atlas Repeat’* mode.

Atlas Repeat:

Using the same 2 x2 atlas for tile patterns that we just used in *Atlas* mode, we change the texture mode to Atlas Repeat. We no longer need the weightings because the atlas is simply 'repeated' across all the tiles.

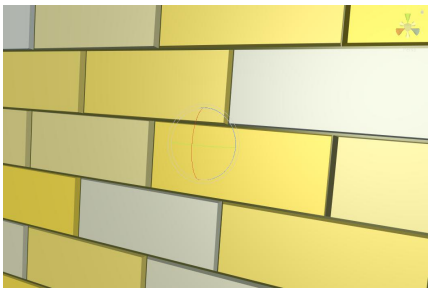


Brick patterning:

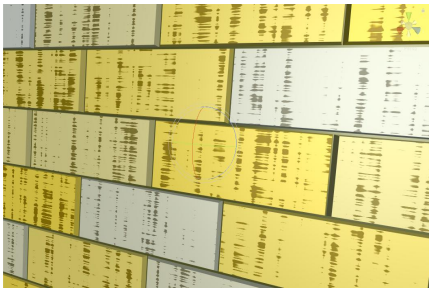
This is the addition of a 'scarring' pattern across the surface of a brick. This typically is either horizontally or vertically dominant. It creates the illusion of wear or abrasion on the surface and is most useful when you are not texturing the surface of bricks as it adds more detail.

The interface provides both a full set of values you can experiment with, as well as a few presets we have worked out. Selecting a preset will set all the values - and then you can fine tune it if you wish, both with the frequencies of the three channels of noise used to create the pattern and also using the '*spread*' control (which controls the 'sharpness' when the range is wider, and the 'amount' by moving the whole range in the slider) and the '*contrast*' which controls how bright the pattern is.

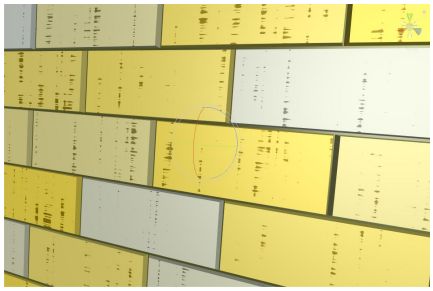
The images below use pale colors to make it easy to see the effect of patterning.



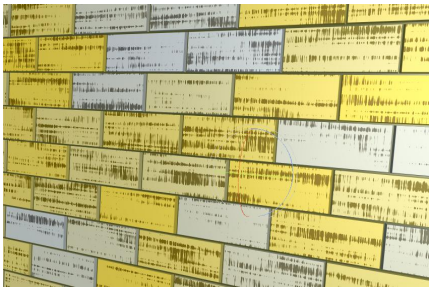
(I) *No patterning*



(ii) Patterning, vertical dominant



(iii) Altering the Spread of (ii)

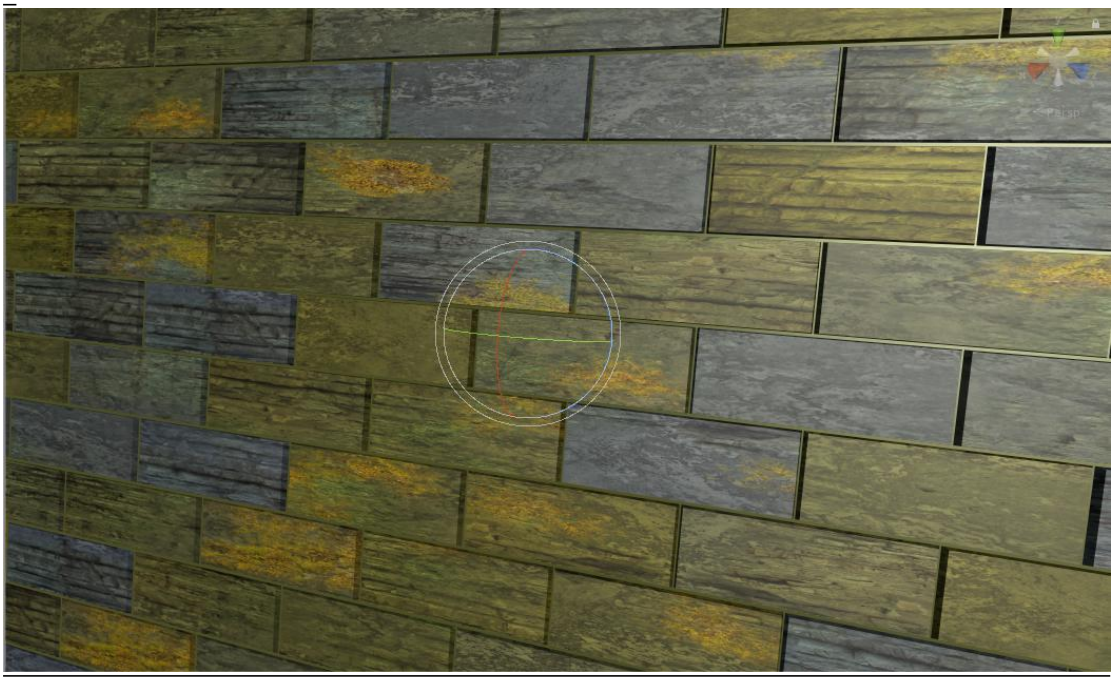


(iv) A more horizontal patterning

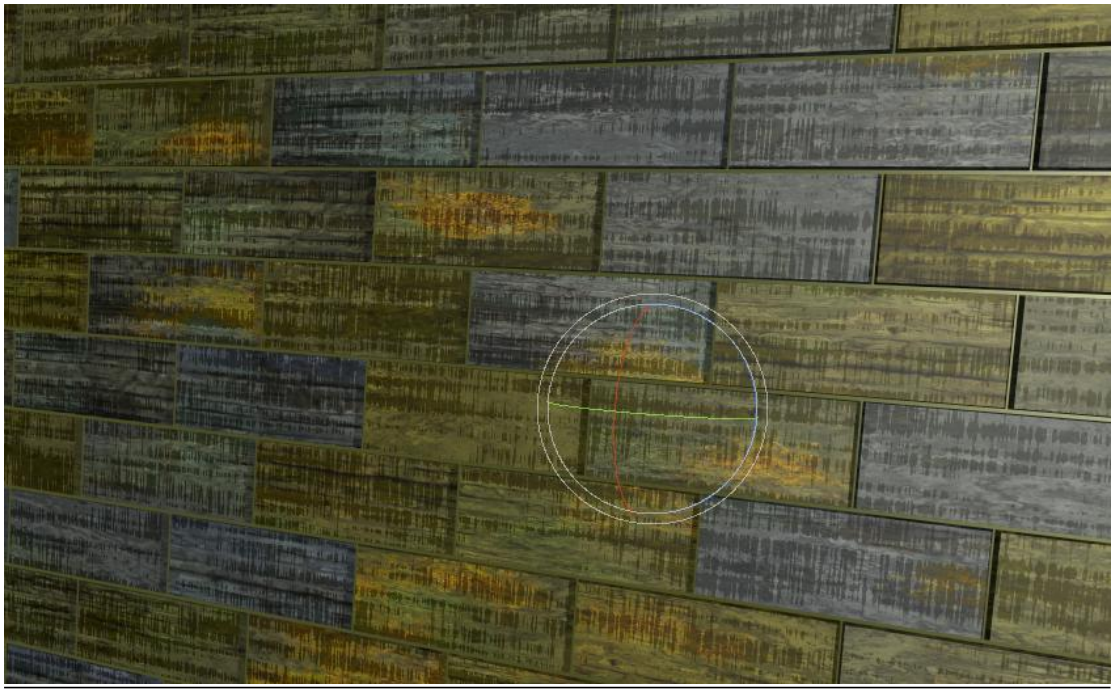
The patterning reduces the intensity of the underlying color wherever the patterning occurs, and the controls ‘Spread’ and ‘Contrast’ let you control the amount of pattern and make it sharper or blurrier. Try experimenting by using the presets and altering the controls to get a feel for this.

As a more representative example, the next two images show how patterning looks when applied to textured brick. Careful use of patterning, particularly on textured bricks, can add the feel of abrasion.

The pattern itself is produced by noise - with independent frequencies for each axis. The higher a frequency, the more ‘detail’ or ‘noisy’ the effect. The channels are all summed together. If a ‘U’ channel frequency is very different to the ‘V’ frequency, the effect is to create horizontal or vertical features. 3 channels is enough to give a good variety. We added some presets in the Interface so that you can just select one and either just use it, or experiment and create interesting patterns.



(without patterning)



(*with patterning)*

Lighting:

Shaders are provided for both simple diffuse and specular lighting (in both opaque and transparent forms).

When using specular lighting you can select to use reflection or not, and whether to use old-style cube maps or the more recent reflection probes. The shader can use either (select from the '*Reflection*' box in the ***Specular*** settings).

Texturing Method:

So far in the document we have used the texture coordinates (UVs) of the mesh as the coordinates for calculating the bricks. However, the shaders support 'automatic' texturing which does not require any texture coordinates. This means that the shader decides how the bricks are to be applied to surfaces, using either the local coordinates or world coordinates.

There are various reasons for wanting to use 'automatic' texturing, including:

- Some scenarios, such as walls made of sections may be difficult to 'match up' bricks at edges.
- World space automatic mode allow meshes to be placed side by side and the brick pattern will tile correctly across the gap.
- Some models may simply not need texture coordinates. The brick pattern may look good using an automatic texturing mode.

Automatic texturing is NOT expensive - it still only does a single sample. This is important to mention since it is NOT TriLinear mapping - (A weighted blend of three samples).

Our automatic texturing does just ONE sample! - since brick typically occurs along flat surfaces (walls!) the shader decides how to calculate the UV's from the space coordinates (Local or World). This means that the shader has no use for texture coordinates with automatic texturing.

With automatic texturing there are several possible ways to shuffle the axes (x,y,z), and you can simply select which one works best - once you apply an automatic mode to your material, simply try each axis permutation from the selection to see which one works best for you!

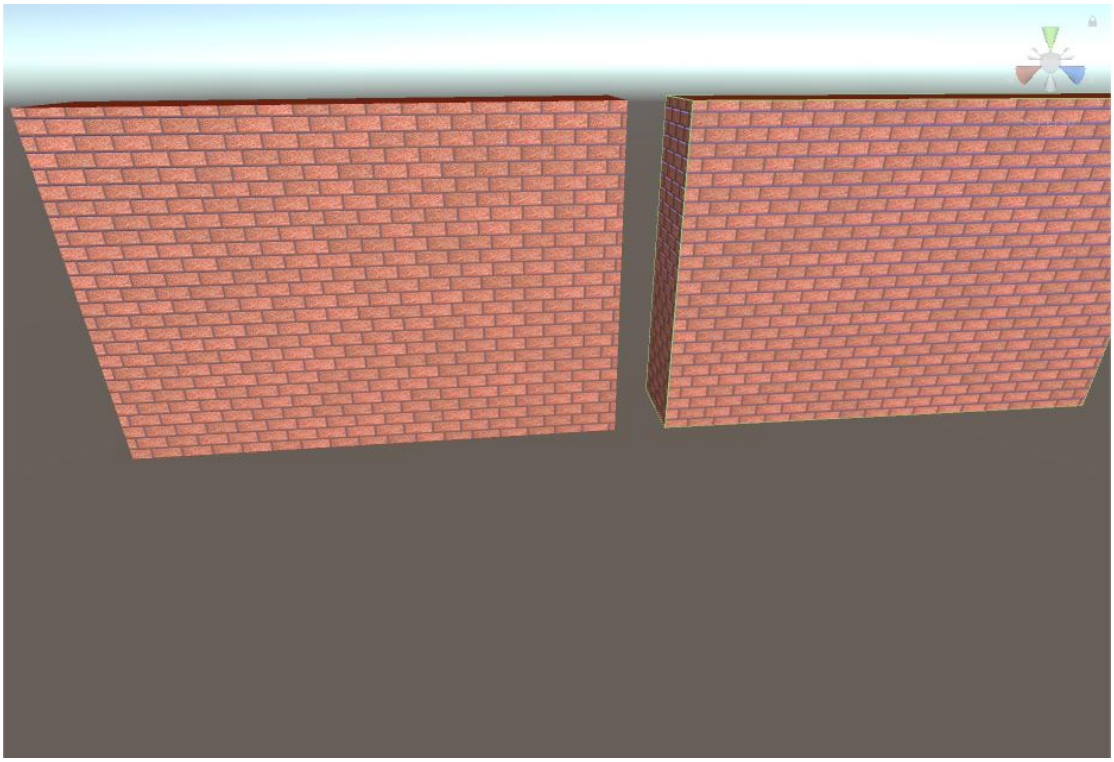
You may find that it takes some experimentation, and since brick walls are often 'axis aligned' you will commonly need to make sure the surface is aligned with world space to look good following example, we create two wall sections, from the standard cube in Unity Editor and we make sure that they are aligned so that we can push them together.

We are using World space so that we can 'join' the two sections seamlessly.

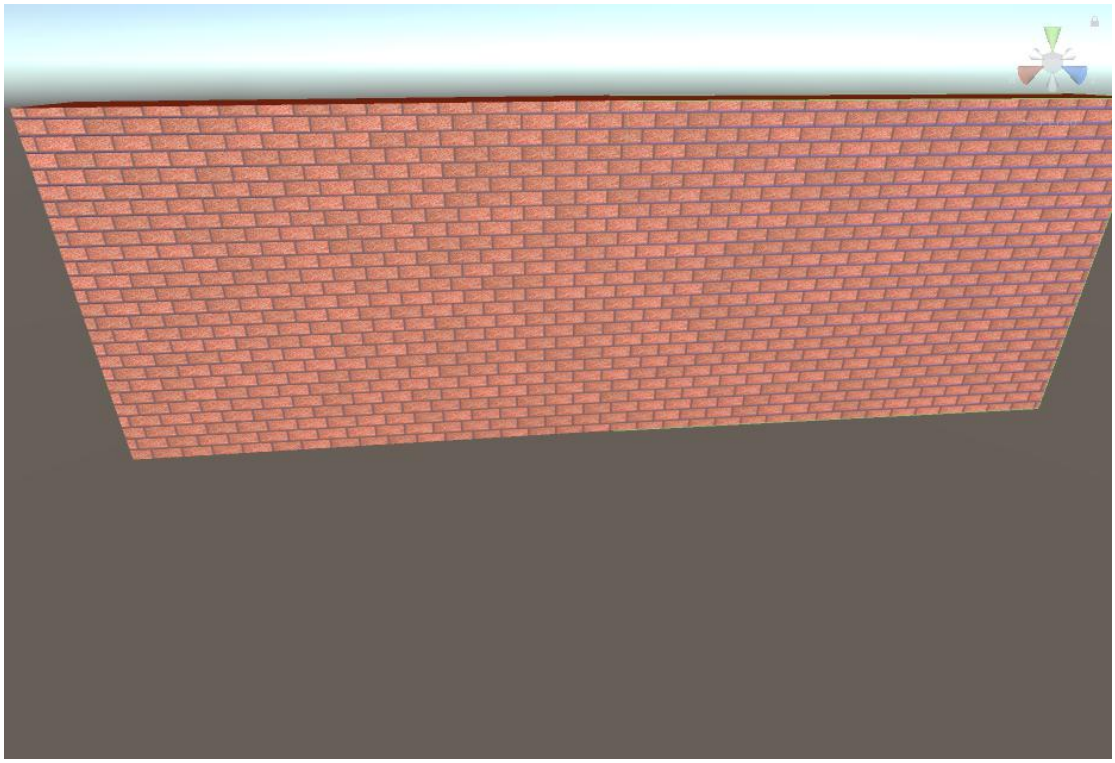


In the first image, the two sections (scaled Cubes, using the same material)
Are seperated by a gap. As we push the right hand cube toward the left hand cube to close the gap, you will see the brick pattern change because it is in world space.

When the two cubes meet, there is no visible join.



Now pushing the two sections together….

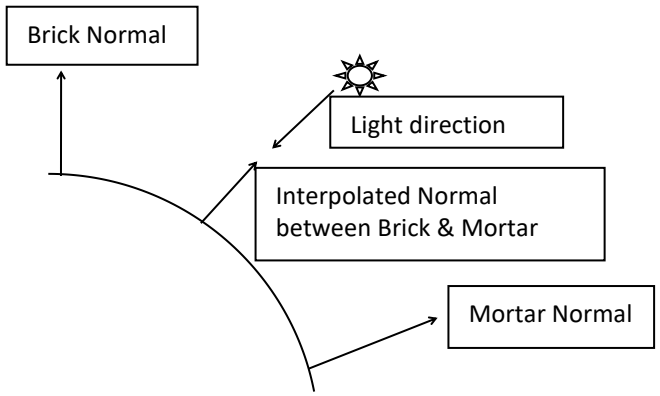


LOD controls:

Level Of Detail (LOD) is needed to prevent artefacts such as 'sparklies' and flickering when you zoom a camera out, and the Interface provides settings which you can use to control how a brick wall will appear when you pull the camera out. The exact values of these settings will depend very much on the colors and lighting you use. Below is a more detailed explanation of how to use these features.

The following is a brief technical note on why we need LOD control in this shader, you can skip this if you just want to go straight to the explanation of using LOD controls.

Why do we even need LOD control ? -
Because the brick geometry is calculated dynamically in the shader, this means that there is no 'overall' texture for the surface. This means we can simply modify the appearance from the interface without having to produce texture maps, normal maps, and so on. However this does mean that we do not have mip-maps to enable filtering, so we have to manage these things ourselves. If we simply interpolated colors and normals at the edges (of brick and mortar, for example) before lighting, then specular highlights would be prone to producing sparkles' which look horrible. We use the differentials (ddx and ddy) of the texture coordinates, to decide how wide an area we need to interpolate across and calculate parametric interpolation factors. These are used in our (custom) lighting routines to interpolate the color of the pixel AFTER lighting.



Getting rid of unwanted sparkles:

At some point the interpolated normal at an edge produces a specular highlight, this is NOT what we want to happen where brick joins mortar, or an edge, so we calculate lighting for each normal, at the pixel, and interpolate those values.

Using The LOD controls.

We find that the easiest way to use LOD settings is a stage at a time. You may find that it is wise to reduce the specular strength of brick and mortar to 0 while you decide to set the 'Fade Distance 'and 'Fade Color', and then set the Specular lighting back and adjust the Specular Merge Rate. This is because it may simply be easier to see the effect of fading color if specular lighting is turned off.

The Fade Color and Fade Distance control the final color to fade the brickwork too. Remember that the bricks and mortar are procedural - ie. Calculated in the shader - so we can't rely on texture mip-maps to do the filtering. Instead we must zoom in and out with the camera in the editor, and set the distance at which the bricks will fade out, choosing a color we like.

Initially set the 'Fade To Distance' slider to 0 - This means that there will be NO fading out to a final color. The final color you want should be set by you in 'Fade Color'. If you are not using any textures (brick or mortar) then the Interface will supply a 'Suggested Color' to you.

The 'Suggested Color' is based on the colors of the brick and the mortar you have selected, as well as the relative areas of brick and mortar - so if you alter the colors, or if you change the relative sizes of brick to mortar (use 'Mortar Gap Across' and 'Mortar Gap Down ' in Brick Sizes to adjust gaps and hence the amount of brick to mortar) then the 'Suggested Color' will change.

If you wish to use the 'Suggested Color', simply click on the button and the Fade Color will be set to the suggested value - you can then adjust it if you want, but we find that usually its a fairly good guess.

If you are using textures for either brick or mortar, then no suggested color is given, you must enter a color which you like.

With a '_transparent' shader you can also set fade levels for the alpha (transparency) to do things like completely fading out walls over distances, and more.

Brick Pattern Fade:

Once you have set the fade color and distance, if you are using Brick Patterning then you will have the option to set the rate at which the pattern fades out when you zoom out. Again its a process of slowly zooming out and when the pattern appears to be slightly flickering carefully increase the Brick Pattern Fade until the pattern is JUST no longer visible. Zoom in and out again to make sure, and do any final tweaking.

Specular Merge Rate:

When using specular lighting this option will be available. Because bricks and mortar may have different specular values set (they probably will!), it means that when you zoom out with camera there comes a point at which flickering patterns may occur. Increase this slider until the specular lighting is uniform and the flickering disappears. This control adjusts the rate at which brick and mortar specular converge to the same values. Again its a process of setting the slider to 0, pulling camera back, and when you just get to the point at which flickering appears, carefully increasing the slider until the flickering stops, then zoom in and out a bit and fine tune if you need to.

It is worth noting that with reasonable specular intensities, the need to fade is not so great as with high values.

Interface (Inspector) controls

This is a comprehensive list of all Interface controls for the procedural brick shader. Some controls will only be displayed for the appropriate shader (for example, there is no need for specular controls when using a diffuse shader), and also some options will only be displayed when the appropriate option is selected (kind of obvious really!).

The meaning of each control is briefly described, for deeper explanation either refer to the first part of this document, or check out the included scenes and experiment.

Optional Settings:

☐ **Brick Texturing** - tick to enable the use of textures.
The following controls are then displayed:

Color - Texture to use for bricks
Note: must have Wrap Mode set to Repeat) in Unity import settings for this texture.
Mode - Atlas / Atlas Repeat / MegaTexture - how to texture
Randomly Flip - Tick to randomlyflip or mirror each brick texture

For Atlas & Atlas Repeat modes only

Bricks Across - Number of brick across (U) in atlas texture
Bricks Down - Number of bricks down (V) in atlas texture

For Atlas mode only

Weighting of brick - likelihood of this brick in atlas being used
(each brick has a 'Weighting of brick' parameter)

Random pattern ID - This value controls the random pattern of tiles. Try values until you find one you like.

For Mega Texture mode only

Brick Scale U - Scale texture in U direction
Brick Scale V - Scale texture in V direction

☐ **Brick Normal Map** - tick to enable the use of a normal map on the brick surface.
The following controls are then displayed:

Normal Map - Normal map texture.
Note: must have Wrap Mode set to Repeat) in Unity import settings for this texture.
If using Brick Texturing it will be sampled with the same texture coordinates. So if you are using an Atlas mode, then the Normal Map must be an atlas too.
Amount Of Normal Map - Slider to set the 'strength' of the normal map

If NOT using brick texture (Brick Texturing is not ticked)

Normal Map Scale U - Scale texture in U direction
Normal Map Scale V - Scale texture in V direction

☐ **Mortar Texturing** - tick to enable the use of a texture on the mortar between bricks, the following controls are then displayed:

Color - Texture to use for Mortar (should tile in both directions)
Mortar Scale U - Scale texture in U direction
Mortar Scale V - Scale texture in V direction

☐ **Mortar Normal Map** - tick to enable the use of a normal map on the brick surface. If using Mortar Texturing it will be sampled with the same texture coordinates.
The following controls are then displayed:

Normal Map - Texture to use for Mortar Normal map
Amount Of Normal Map - Slider to set the 'strength' of the normal map

If NOT using mortar texture (Mortar Texturing is not ticked)

Normal Map Scale U - Scale texture in U direction
Normal Map Scale V - Scale texture in V direction

☐ **Brick Pattern** - tick to enable the use of procedural patterning on the brick surface, the following controls are then displayed:

Brick Noise Presets - Several presets displayed, select by clicking radio button.

Freq 0 Across (U) - Channel 0 noise frequency in U direction
Freq 0 Down (V) - Channel 0 noise frequency in V direction

Freq 1 Across (U) - Channel 1 noise frequency in U direction
Freq 1 Down (V) - Channel 1 noise frequency in V direction

Freq 2 Across (U) - Channel 2 noise frequency in U direction
Freq 2 Down (V) - Channel 2 noise frequency in V direction

Frequency Multiplier - An overall control of ALL frequencies.
(Smaller values = bigger patterns)

Spread - softens (length of range) & amount of pattern.
(Experiment to understand this properly).

Contrast - How 'bright' the patterning appears on the brick

☐ **Brick Raised Edges** - Tick to enable the use of raised edges on the brick surface, the following controls are then displayed:

Edge Type - Rounded or Flat, the shape of brick/tile edges.
Distinct Edge Texture - Tick box, select this for a different texture on edges.

If Distinct Edge Texture is selected

Edge Color - Texture to apply to edge of brick

Elevation - Make the edge seem steeper or shallower, use in Conjunction with edge thickness controls.

Thickness Right - Thickness of right edge
Thickness Left - Thickness of left edge
Thickness Lo - Thickness of lower edge
Thickness Hi - Thickness of higher edge

(Note that with rotation etc. of objects, and UVs, that these sides may appear to be opposite in some situations, eg. Left and Right swapped).

If Edge Type is set to 'rounded'

Corner Rounding - The 'rounded'ness of corners of bricks.
Slider goes from: 0 (sharp) to 1 (smooth)

General Settings:

▼ ***Specular*** - Select to open up the following settings for Specular lighting

Reflection - None / Cube Map / Probe. How to handle reflection in Specular lighting.

None = No reflection
Cube Map = Old style Cube Map needed for reflection.
Probe = Use Reflection Probes from scene.

If Reflection is Cube Map

Reflection Cubemap - Old style cubemap used for specular reflection.

The specular settings for brick and mortar now follow. Strength is how bright the highlight is. Power controls the size of the highlight (smaller = bigger highlight area).

Brick

Specular Power - Bigger values = smaller specular highlight
Specular Strength - Intensity of specular highlight
Reflection Strength - How much reflection to apply for highlight

Mortar

Specular Power - Bigger values = smaller specular highlight
Specular Strength - Intensity of specular highlight
Reflection Strength - How much reflection to apply for highlight

▼ ***Colors*** - Select to open up the following Color settings
Note: When using textures, colors tint the textures.

Brick Tint Lo - Lo end of range of colors for brick tint
Brick Tint Hi - Hi end of range of colors for brick tint
Edge Tint - Color of Edge
Mortar Tint - Color of mortar
Albedo Tint - Albedo overall for lighting (normally white)

▼ ***Brick Sizes-*** - Select to open up the following settings for Brick Sizes

Use Tiling factors to determine the desired brick size
(see document above - in section: ***Setting the size of bricks and mortar***)

Mortar Gap Across - Amount of mortar. In U direction (across)
Mortar Gap Down - Amount of mortar. In V direction (down)
Tiling Scale U - U repeat factor, higher = more bricks across surface
Tiling Scale V - V repeat factor, higher = more bricks down surface
Overall Scale - scale both U and V together
Tiling Offset U - U offset
Tiling Offset V - V offset

Indent Axis - Which axis (U or V) to apply an indent amount to
Indent Odd - Indent amount for ODD rows
Indent Even - Indent amount for EVEN rows

▼ ***Texturing Method*** **-** Set the way texture coordinates are sourced

Usually you will have this set to 'Texture Coords' which means the shader uses the model's UVs. However the shader supports 'automatic texturing' Read the section ***'To Texture or Not to Texture'*** *in the document above.*

UV Source **-** Where to get UV's from -

Texture Coords = models texture coords.
Auto Local Space = local space coordinates
Auto World Space = world space coordinates

If using an automatic mode. The following option is available

Shuffle Axes - Which axis permutation to use for automatic texturing (just try them out if you use an automatic mode to find which works best in your scenario).

▼ ***LOD controls*** **-** Select to open up the following settings for Level Of Detail control (Please read the section on LOD controls earlier in this document)

Fade To distance - controls how far away the camera is when bricks & mortar become the fade color.

If you are NOT using textures then the interface suggests a fade color and the following controls are given.

Fade Color (Brick to Mortar) - The amount of influence brick or mortar have on the suggested fade color.
The slider ranges from fully Brick to fully Mortar.

Suggested Color - The color which the Interface thinks will look good as a fade color (see the notes in the document above - In Section **LOD Controls**). Choose this color by clicking on the radio button - the color will be copied into the Fade Color control (below).

Fade Color - The color to fade to. If you are using brick or mortar textures then you have to select a color manually. If you are NOT using textures Then the editor suggests a color (this will be displayed as 'Suggested Color' above) which you can select if you wish..

Fade Alpha Brick - The level of transparency to fade to for bricks.
- This lets you fade out bricks over distance when Using a '_transparent' shader.

Fade Alpha Mortar - The level of transparency to fade to for mortar.
- This lets you fade out mortar over distance when Using a '_transparent' shader.

Specular Merge Rate - The rate at which specular highlights converge to a uniform value.

If using Brick Patterns

Brick Pattern Fade - Set rate at which patterning fades away with distance.

And finally:

The example scenes and assets included with these shaders are there for you to experiment with, have fun!

Support notes:

For any technical enquiries email **brynderitech@gmail.com**