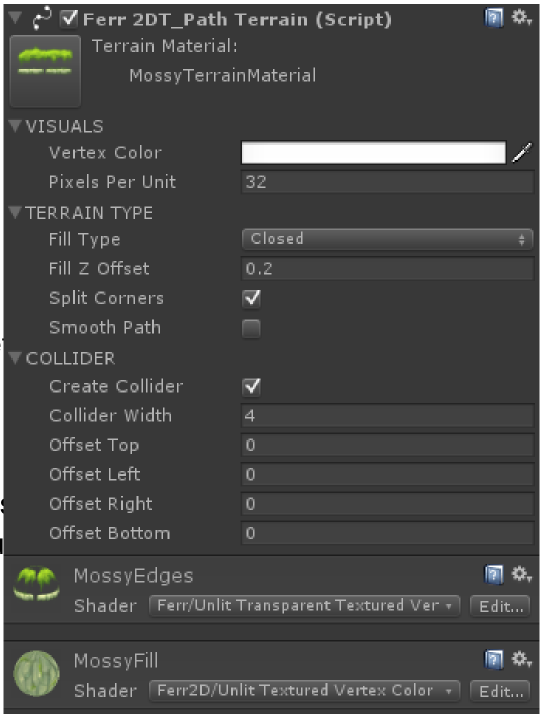
Reference

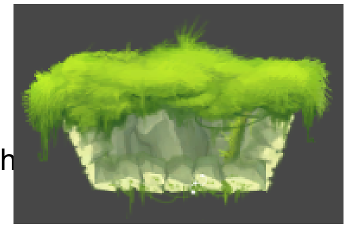
Ferr2DT_PathTerrain

This is the primary component of concern for the tool, and will probably be what you spend the most time with, so it's a good one to understand well. This component relies on data from the Ferr2D_Path component, so if you're looking for the path itself, you may wish to check there.

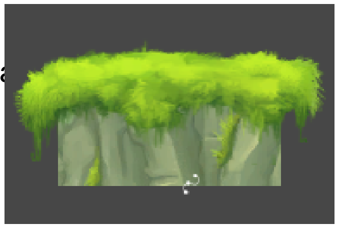


Ferr 2DT_Path Terrain (Script)
Terrain Material: MossyTerrainMaterial
VISUALS
Vertex Color
Pixels Per Unit 32
TERRAIN TYPE
Fill Type Closed
Fill Z Offset 0.2
Split Corners ☑
Smooth Path ☐
COLLIDER
Create Collider ☑
Collider Width 4
Offset Top 0
Offset Left 0
Offset Right 0
Offset Bottom 0
MossyEdges
Shader Ferr/Unlit Transparent Textured Ver Edit...
MossyFill
Shader Ferr2D/Unlit Textured Vertex Color Edit...

- ***Terrain material***
  - The terrain material can always be changed at a moment's notice! This brings up the terrain material selection window that you see when creating a new terrain.
- ***VISUALS***
  - **Vertex Color**
    - The vertices of the mesh are se to this color, this allows you to tweak and tint your terrain for lighting and variety while still letting Unity batch your materials for performance! Make sure you use a shader that supports vertex colors to take advantage of this.
  - **Pixels Per Unit**
    - The easiest measure for ensuring consistent pixel density. How many pixels should the tool attempt to fit into one unit of Unity space? This isn't always exact, but it's a target that the tool strives towards.
- ***TERRAIN TYPE***
  - **Fill Type**
    - **Closed**

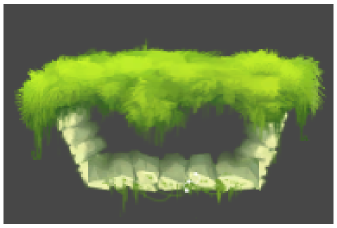


      - The interior of the path is filled, and edges go the entire way around the path including the closing segment.
    - **Skirt**

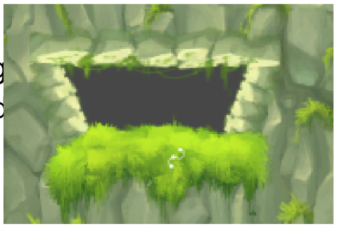


      - The interior of the path is filled, and extra 'skirt' points are dropped down in order to close the terrain shape.

        - **None**
            - The interior will not be filled, only edge materials will be applied. The closing segment will only be present if the accompanying Ferr2D_Path component is marked as ‘closed’.



        - **Inverted Closed**
            - The exterior of the path is filled, and edg directions are flipped to match an inward facing direction. This is excellent for creating interiors.



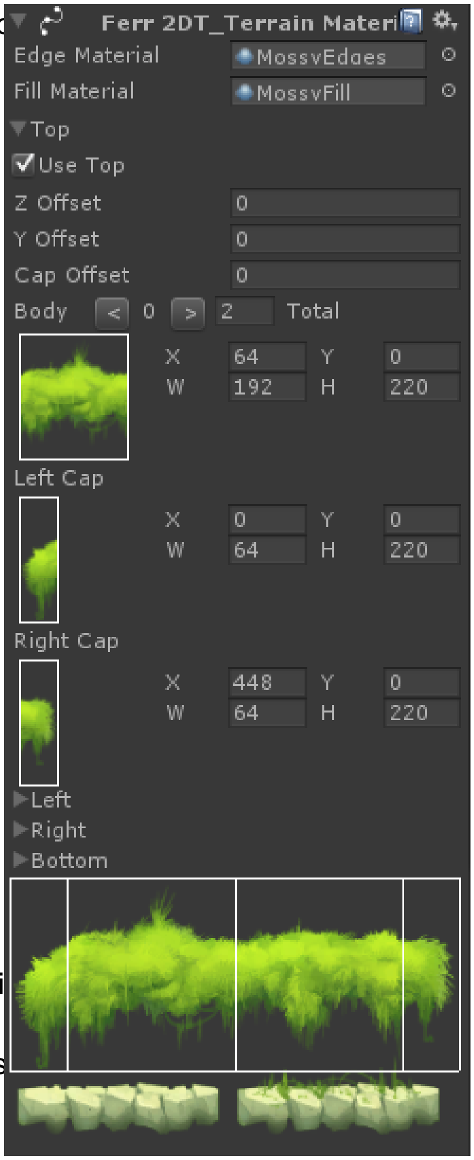
    - **Fill Z Offset** (Fill type != None)
        - In order to avoid Z-fighting issues, it is often best to offset the fill mesh by a small amount. You can tweak this amount here, but the default should normally be acceptable.
    - **Skirt Y Value** (Fill type = Skirt)
        - When fill type is set to Skirt, this is the Y value that the skirt will drop to. This value is in local space, not global.
    - **Split corners**
        - When true, this will break any corners into new segments. This is primarily used to apply different materials to segments facing separate directions. If you are using a terrain material with only one edge defined, this should be turned off.
    - **Smooth path**
        - This feature is still in progress, but it will attempt to smooth the terrain segments for a more gentle curve. The current implementation adds many extra path points behind the scenes, which can have a significant performance impact on the triangulation algorithm. This should be fine for terrain objects with a fill type set to ‘None’
- ***COLLIDER***
    - **Create Collider**
        - When checked, the tool will create a MeshCollider on Start. It will also display a green line in the Scene window, indicating the shape of the collider.
        - For a video on working with colliders, check here: http://youtu.be/vLpzLyONFGY
    - **Collider Width**
        - This tells the extent of the collision mesh along the Z axis, in Unity units.
    - **Thickness Top** (Fill type = None)
        - An offset from the path for the top side of the collider.

- **Thickness Bottom** (Fill type = None)
  - An offset from the path for the bottom side of the collider.
- **Offset Top** (Fill type != None)
  - An offset from top edge segments for the collider.
- **Offset Left** (Fill type != None)
  - An offset from left edge segments for the collider.
- **Offset Right** (Fill type != None)
  - An offset from right edge segments for the collider.
- **Offset Bottom** (Fill type != None)
  - An offset from bottom edge segments for the collider.

Ferr2DT_TerrainMaterial

The Terrain Material describes how materials are applied tc the Path Terrain. You can define a single edge using the "top" edge in order to achieve things like pipes and branches, or you can define all 4 to create a regular terrain material. Check out some of the example materials to see what sort of things you can create with it!

For video tutorials of this, check out these:
http://youtu.be/_V9v2RXkUog
http://youtu.be/TYQSuhsyCDE


Ferr 2DT_Terrain Materi
Edge Material: MossyEdges
Fill Material: MossyFill
▼Top
☑ Use Top
Z Offset: 0
Y Offset: 0
Cap Offset: 0
Body < 0 > 2 Total
X 64 Y 0
W 192 H 220
Left Cap
X 0 Y 0
W 64 H 220
Right Cap
X 448 Y 0
W 64 H 220
▶Left
▶Right
▶Bottom

- **Edge Material**
  - This is the material that will be used by the tool for all edge pieces of the terrain. The included shader, "Ferr/Unlit Textured Transparent Vertex Color" was designed for this role, but it is not essential for the terrain to work correctly.

    For most edge materials, a shader that supports transparency and vertex color should take advantage of all the tool's features.
- **Fill Material**
  - This is the material that will be used by the fi mesh. The included shader, "Ferr/Unlit Textured Vertex Color" was designed for this role, but it is not essential for the terrain to work correctly.

We recommend against using transparent materials for this, as transparent materials can be quite expensive, especially on mobile.

- **(Top, Left, Right, Bottom)**
  - **Z Offset**
    - Z fighting will occur often when creating terrain materials edges, so in order to avoid it, it is best to offset the edge mesh by a small amount.
  - **Y Offset**
    - This will offset the edge mesh from the path by a certain amount. This can be extremely helpful for lining up edge materials better.
  - **Cap Offset**
    - A work in progress, this will allow you to adjust where the cap meshes are at the end of their segments. This can also help with lining up edge materials. Beware, the current algorithm can sometimes end up squishing the texture, especially on smoothed terrain paths.
  - **Body**
    - The “body” of this edge is the segment of the Edge Material that will be repeated along the bulk of the path.
    - X,Y,W, and H are measured in pixels from the top left corner of the texture. These are stored as UV coordinates internally, so if you resize the texture, they will resize with the texture.
    - By increasing the “Total” variable, you may have multiple body segments for the tool to randomly choose from when generating the mesh.
    - Use the < and > arrows to pick between the different body segments available to the tool.
  - **Left Cap, Right Cap**
    - This is a small material segment that is placed at the end of each edge to cap it off. These are optional.

Others

There are more code and components available for use, however, these components are either placeholder components to get you started, or they are meant to help with code and are well comment-doc’d for use with intellisense. Here’s a few things you might find interesting!

- **Ferr2D_Path**
  - This represents a basic 2D path. It has a ton of helper functions in it for path manipulation and smoothing.
- **Ferr2DT_Triangulator**
  - This class will turn a collection of points into a 2D mesh, and is used to create the fill meshed for the terrain. However, there’s always plenty of other additional uses for something as useful as that!
- **Ferr2D_Sprite**
  - A really basic 2D sprite component that I’ve used for UI and basic 2D things in the past. It’s not designed to really hold its own, but it’s an acceptable placeholder if

you have nothing.

- **Ferr2D_Animator**
  - A basic grid based spritesheet animation component. Again, not really designed to hold its own, but works as a decent placeholder if you have none. Features multiple animations, auto animation transitioning, callbacks, and an inspector interface.
- **Ferr2DT_DynamicMesh**
  - This one’s a little tool specific, and I don’t recommend direct use. Next update, I should have a much better mesh utility class that will work in generic cases.