Super Tilemap Editor Manual

Creative Spore

Support: creativespore@gmail.com

Web: http://www.creativespore.com

Unity Forums: user CreativeSpore

Index

1 Introduction

This is the user manual for the Unity asset Super Tilemap Editor.

Super Tilemap Editor is a fast and easy to use tile editor to help you to create any game based on tiles.

This manual will teach you how to use and configure the tilesets and tilemaps to use them properly in your projects and created tiled games really easy and fun.

2 Getting Started

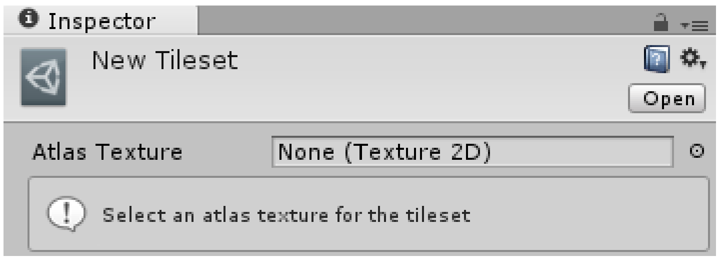
If you are really anxious to use the tilemap editor and have no time for reading all the manual, this will give you a fast introduction to start creating tilemaps for your games.

2.1 Creating a Tileset

Before creating a tilemap in the scene, you need to create a tileset to be used by the tilemap.

The tileset is basically composed by a texture atlas with the tiles, a list of the tiles sliced from the atlas and brushes (they will be explained later).

To create a tileset, go to "**Assets/Create/SuperTilemapEditor/Tileset**".

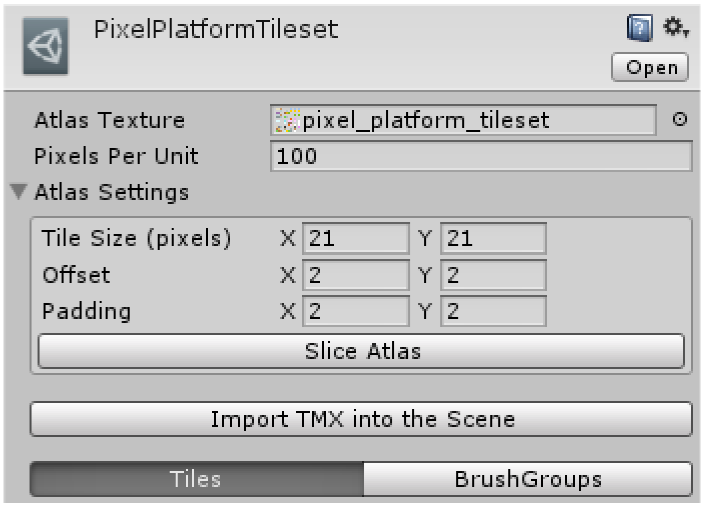


Atlas Texture None (Texture 2D)

Select an atlas texture for the tileset

The tileset will be created without an atlas texture selected.

Let's select one. I am going to select the one included with the asset based on a Kenney's free to use texture.



PixelPlatformTileset

Atlas Texture: pixel_platform_tileset
Pixels Per Unit: 100
Atlas Settings
Tile Size (pixels) X 21 Y 21
Offset X 2 Y 2
Padding X 2 Y 2

Now, we can see new options: Grid and Tile Palette.

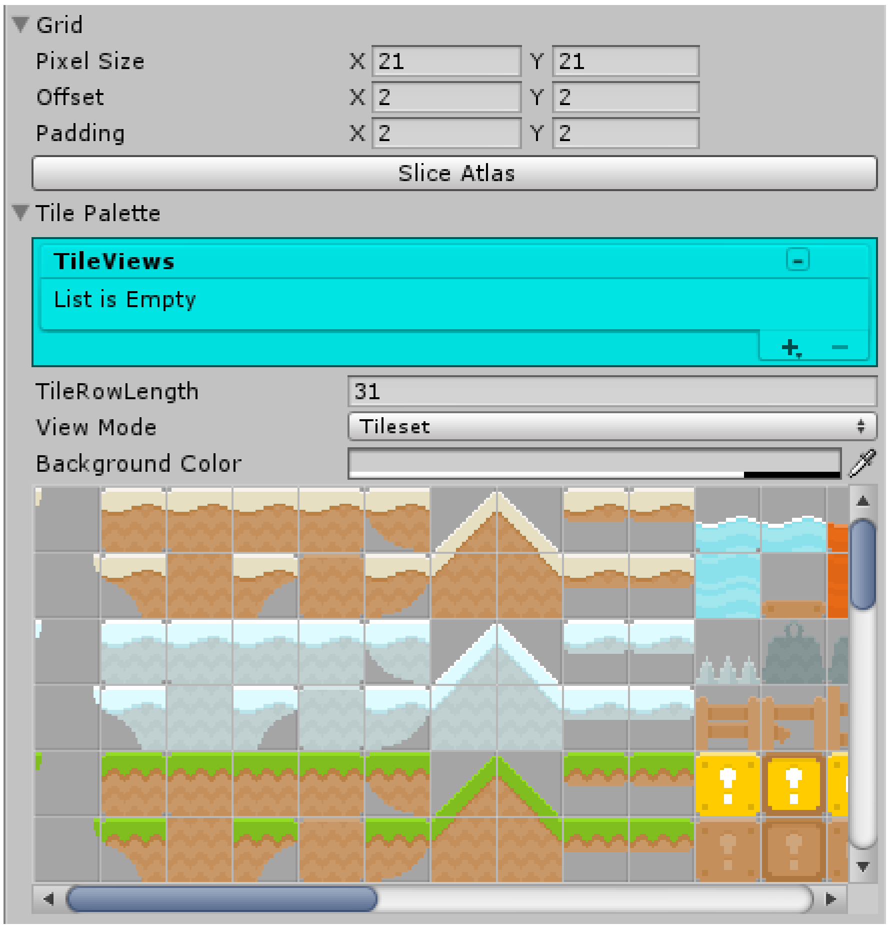
But we still need to slice the atlas to create the actual tiles used by this tileset.

Pixels per Unit will take this value from the Atlas Texture, but you can change it here to a different value. When a tilemap is created, this value will be used to set the cell size of the tile, but you can change the cell size as well after the creation of the tilemap.

Slicing the atlas works the same as Unity Sprite Editor:

- **Tile Size**: is the size of the tile in pixels
- **Offset**: is an offset in pixels to start slicing the texture
- **Padding**: is the separation between tiles in pixels

Once we set the parameters, press "Slice Atlas" and you will see the tile palette. If you miss a parameter, fix it and slice again.



Grid

Pixel Size X 21 Y 21

Offset X 2 Y 2

Padding X 2 Y 2

Tile Palette

TileViews

List is Empty

TileRowLength 31

View Mode Tileset

Background Color

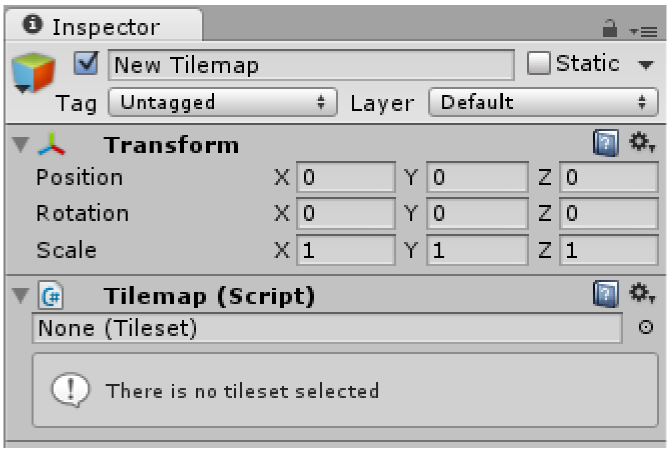
And that's it. You have finished creating your first tileset.

Let's now create a tilemap using your new tileset.

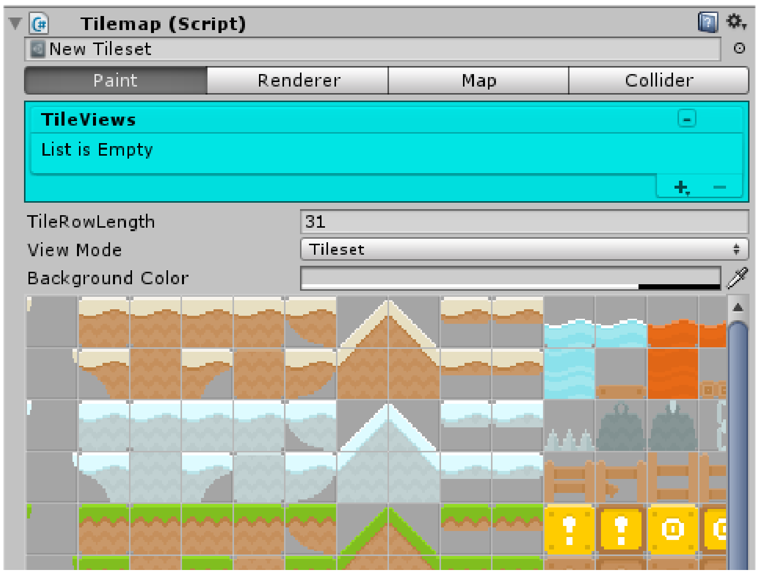
2.2 *Creating a Tilemap*

Tilemaps are gameobjects created in a scene.

To create a tilemap go to "**GameObject/SuperTilemapEditor/Tilemap**".

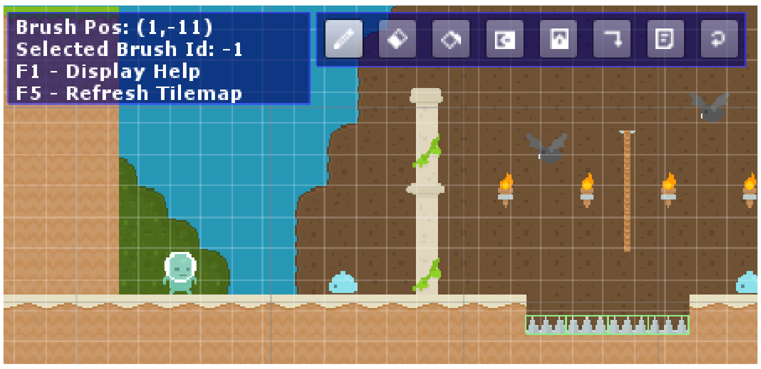


The tilemap need a Tileset to work. Let's select the tileset created in previous section "Creating a Tileset".



Once a tileset has been selected, a list of options appear.

To paint in the scene tilemap, you have to select the **Paint option**, then press left click over any tile in the tile palette and start painting in the scene by left clicking.



You can select more than one tile from the tile palette by dragging the mouse holding left button.

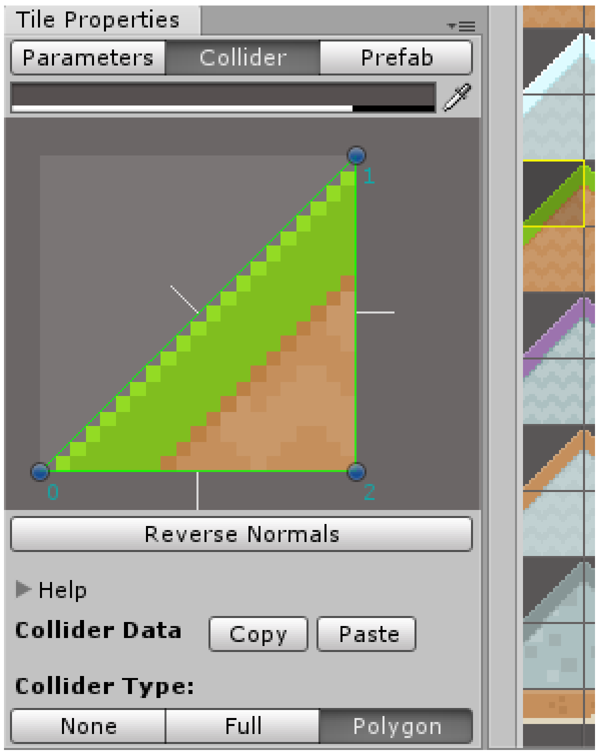
Hold center or right mouse button and drag to move the tile palette scroll.

While painting in the scene, you can:

- Paint holding left mouse button
- Erase holding shift + left mouse button
- Fill an area by double clicking with a single tile selected
- Copy tiles by dragging and holding right mouse button
- Cut tiles by dragging and holding right mouse button and Shift key
- Select a tile or brush by right clicking over the tile. If it's a brush, the brush will be selected first and second time the tile painted by the brush will be selected cycling between them.
- Rotate and flip the brush tiles by using special keys:
    - Rotate 90º by using comma ',' and period '.'
    - Vertical Flip by pressing X
    - Horizontal Flip by pressing Y
    - Hold shift to only rotate or flip tile positions
- Use Ctrl-Z and Ctrl-Y to undo/redo changes

2.3 *Creating tile colliders*

To modify the collider properties of a tiles you have to open the tile properties window. Go to "**Window/SuperTilemapEditor/Tile Properties Window**".



Select Collider section in the tile properties window.

You will see the tile selected in the tile palette.

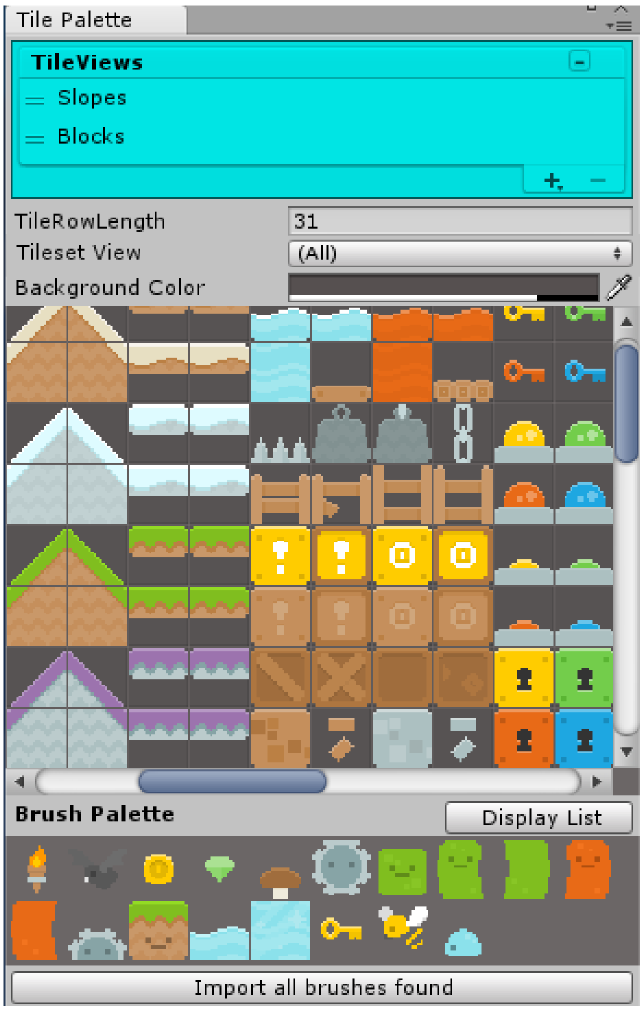
It could be None, Full or Polygon.

Trick: right click over a tilemap tile to select the tile faster from your level.

The normal lines in white will show the collision side. It depends of the direction of the vertices. You can flip the normals by pressing "Reverse Normals" button.

3 Super Tilemap Editor

3.1 *Tile Palette*



TileRowLength 31
Tileset View (All)
Background Color

The Tile Palette can be found in different places like:

- The tileset inspector view
- The Tilemap inspector view with Paint tab selected
- The Tile Palette Window "**Window/SuperTilemapEditor/Tile Palette Window**"

This control is connected with a source tileset. In case of the Tile Palette Window, the source tileset will be set to the last selected tileset or tilemap (in this case, the tileset used by the tilemap).

Here are displayed the tile views (a custom selection of tiles), all the tileset tiles availables, and all the brushes found using the source tileset.

There are some settings you can change for a better display of the data:

- **TileRowLength**: this is set by default to the tileset maximum row length and specify the number of tiles per row. The tiles will be wrapped to fit this number.
- **Tileset View**: here you can select (All) to display the full palette or a created tile view.
- **Background color**: change the color of the background.

From the tile scroll view, you can select one tile or a rectangle selection of tiles to be painted on a tilemap or for creating a new tileview.

Press and hold left mouse button to make a rectangle selection or just click over a tile to select it.

Press and hold right or center mouse button to move the tile scroll view.

You can also select a brush in the bottom brushes palette. Brushes will be explained in the brushes section.

Double click over a brush will ping the brush asset in the project view and show its properties in the inspector view.

Right click over a tile to display its properties.

3.1.1 Tile Views

Tile Views are selection of tiles saved as a tile view to display only that specific selection.

To create a tile view, make a rectangle selection of tiles, then press the '**+**' button and select "**Add Tile Selection to TileView**" to add a selection from the tile palette or "**Add Brush Selection**" to add the brush selection.

To remove a tile view, select the tile view and then press the '**-**' button.

Selecting a tile view will automatically display the tile view in the tile palette.

You can rearrange tile views as you wish by pressing and dragging the tile view in the list.

To see all tiles again, change the **View Mode** to Tileset.

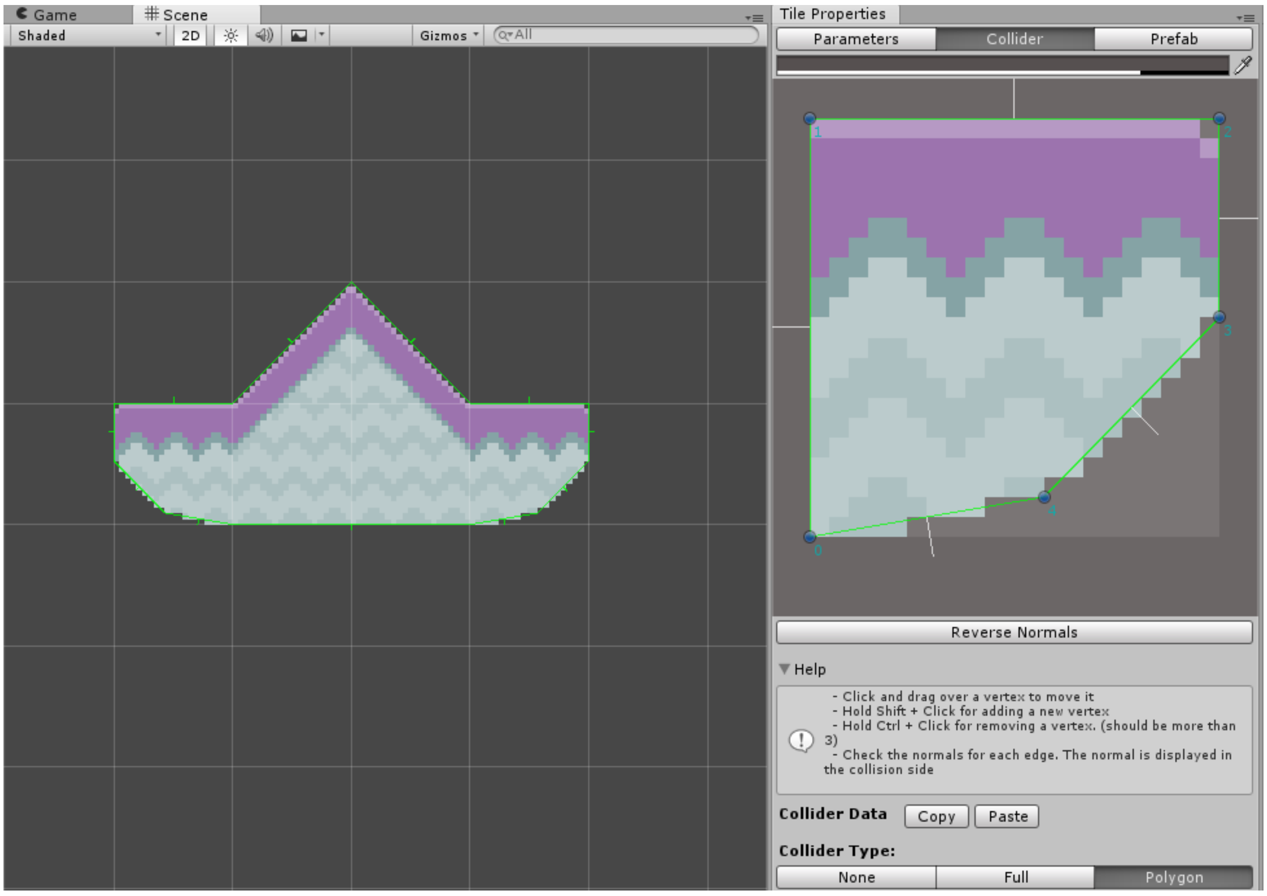
3.2 Tile Properties

The tiles properties are displayed by the Tile Properties Window "**Window/SuperTilemapEditor/Tile Properties Window**".

Here will be displayed the properties of the current selected tile. Select a tile using any tile palette. In case of multi-selection only one of the tiles will be displayed.

These properties are divided in Colliders and Parameters.

3.2.1 Tile Properties - Colliders



- Click and drag over a vertex to move it
- Hold Shift + Click for adding a new vertex
- Hold Ctrl + Click for removing a vertex. (should be more than 3)
- Check the normals for each edge. The normal is displayed in the collision side

Collider Data: Copy | Paste

Collider Type: None | Full | [✓ Polygon]

In this section you can set the collider type of the tile:

- **None**: the tile has no collider
- **Full**: the full tile is a collider
- **Polygon**: custom polygon collider

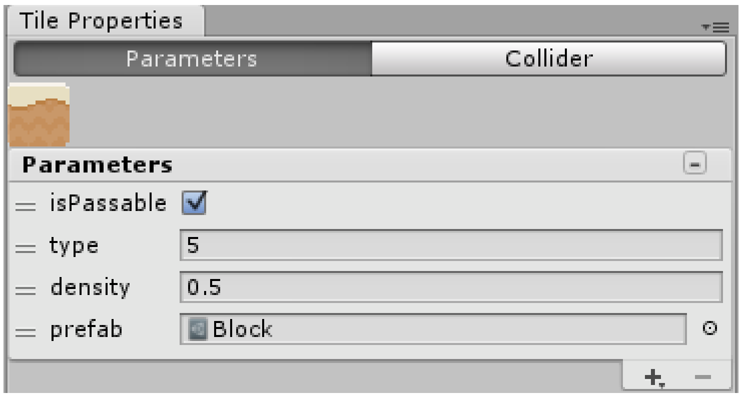
Only the polygon collider allows to create, remove and modify the collider vertices:

- Click and drag over a vertex to move it
- Hold Shift + Click for adding a new vertex (mouse should be over an edge)
- Hold Control + Click for removing a vertex (with a minimum of 3 vertices)

You can copy the tile collider data of the current displayed tile and paste it selecting another tile or selection of tiles (**Multi-selection edition** will be displayed in that case and Paste will modify all of them with the previous copied data)

The normal of each collision edge is shown with a perpendicular line in the facing direction. Normally the normal should be facing the outside. You can press **Reverse Normals** to make them face the opposite direction.

3.2.2 Tile Properties - Parameters



isPassable ✓
type 5
density 0.5
prefab Block

Each tile in the tileset, and brush assets can have a list of parameters of type int, float, bool and object.

These parameters can later be accessed by code for custom logic, like creating an instance or set special properties.

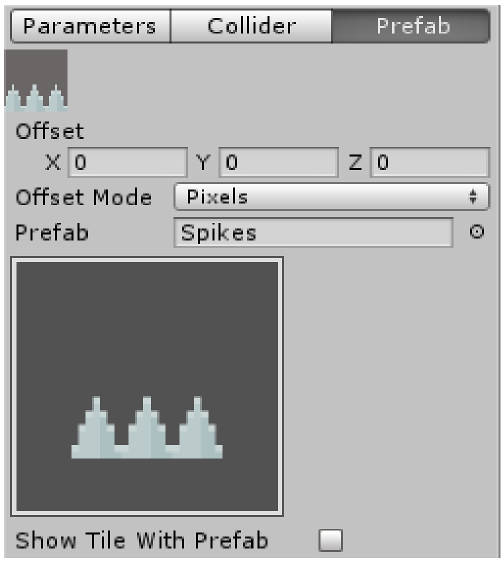
To access the parameters of a tile, select the tile or brush from any tile palette (Tile Palette Window, Tilemap\Paint ot Tileset).

To add a new parameter, press the '**+**' buton, and the '**-**' button to remove a parameter.

You can rename any parameter by clicking over its name and drag parameters to rearrange them.

```
// If parameter  "paramName" exists, sets the value to true
Tiles[0].paramContainer.FindParam("paramName").SetValue(true);
// Creates a new integer parameter named "paramName". To create a float parameter use 25f
Tiles[0].paramContainer.AddParam("paramName", 25);
// "paramName" should exist and be an integer. Takes the value of this integer parameter
Tiles[0].paramContainer.FindParam("paramName").GetAsInt();
```

3.2.3 Tile Properties - Prefab



Offset
X 0 Y 0 Z 0
Offset Mode: Pixels
Prefab: Spikes
Show Tile With Prefab: ☐

A tile can create a prefab instead of being painted if a prefab is set in the **Prefab** data.

You can set an **offset** for this prefab, centered in the center of the tile.

The offset could be in units or pixels changing the **Offset Mode**.

If the offset is in pixels, the offset will keep the aspect ratio even if the tilemap cell size is changed later.

Show Tile With Prefab set if the tile should be shown or hidden when it has a prefab attached.

3.3 Tilemap

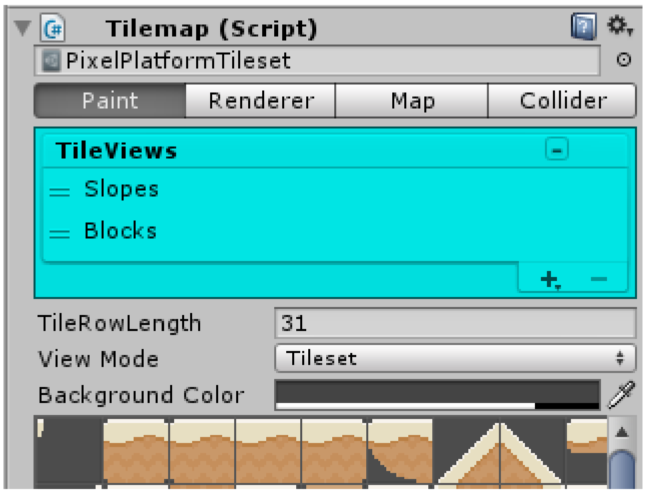
Tilemaps are the game objects containing the map of tiles. Each tilemap have an unique tileset for drawing tiles.

Imagine the tilemap like an infinite plane where you can draw using the tile brush.

The transform properties can changed, to rotate, scale and move the tilemap plane over the 3D space.

The tilemap inspector shows different options for different setting of the tilemap.

3.3.1 Tilemap - Paint


Tilemap (Script)
PixelPlatformTileset

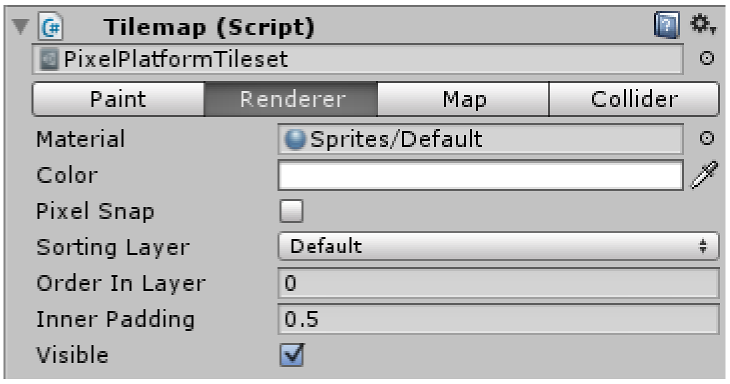
TileViews
Slopes
Blocks
TileRowLength 31
View Mode Tileset
Background Color

This tab should be selected in order to paint tiles on the selected tilemap.

The inspector will show the tile palette where you can access the different tile views, select a tile or a rect of tiles or select a brush.

To know more about painting tiles, check the section #2.2.Creating a Tilemap.

3.3.2 Tilemap - Renderer



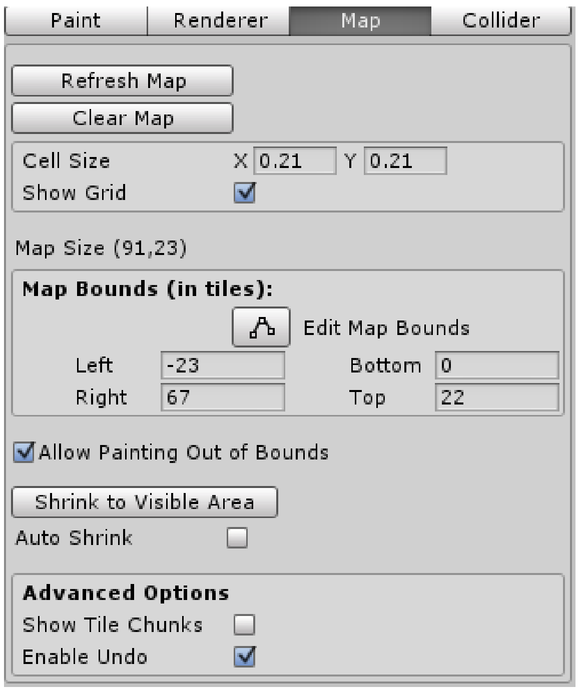
PixelPlatformTileset

Material	Sprites/Default
Color	
Pixel Snap	☐
Sorting Layer	Default
Order In Layer	0
Inner Padding	0.5
Visible	✓

In this section you can change the visual aspects of the tilemap:

- **Material**: the material used to render the tile mesh
- **Color**: the tint color for the tilemap
- **Pixel Snap**: pixel perfect option, only if Sprite material is used
- **Sorting Layer**: the sorting layer used by the tilemap
- **Order In Layer**: the order inside the selected layer
- **Inner Padding**: the size, in pixels, the tile UV will be stretched. Use this to fix pixel precision artifacts when tiles have no padding border in the atlas.
- **Visible**: enable/disable the render of the tilemap

3.3.3 Tilemap - Map



Cell Size X 0.21 Y 0.21
Show Grid ✓

Map Size (91,23)

Map Bounds (in tiles):

Left	-23	Bottom	0
Right	67	Top	22

✓ Allow Painting Out of Bounds

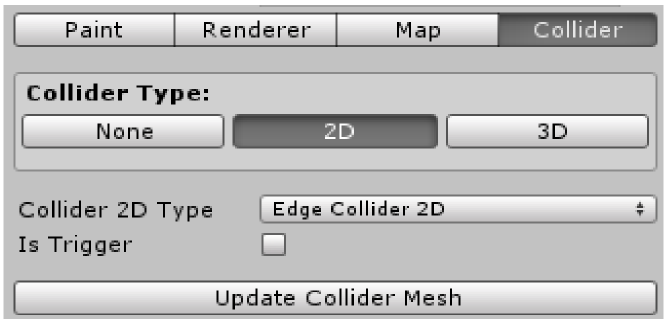
Auto Shrink ☐

Advanced Options
Show Tile Chunks ☐
Enable Undo ✓

This section display properties relatives to the tilemap:

- **Refresh Map**: will refresh the whole tilemap, tiles and colliders. Use this button when changing tile properties like colliders to update the map with the changes.
- **Clear Map:** will remove all tiles from the tilemap.
- **Cell Size**: the size of the cell where the tile is drawn.
- **Show Grid**: show/hide the tile grid.
- **Map Size**: this is the size of the map, automatically increased to enclose all drawn tiles.
- **Map Bounds:** this is the limit area of the map. It is automatically updated while painting outside the map bounds but you can manually set the bounds here or press the **Edit Map Bounds** button and drag the handles directly in the scene view.
- **Allow Painting Out of Bounds:** is unchecked it would be only possible to paint tiles inside the map bounds.
- **Shrink to Visible Area:** reduce the map bounds until fit all visible tiles.

3.3.4 Tilemap - Collider



Collider Type:

Collider 2D Type: Edge Collider 2D

Is Trigger: ☐

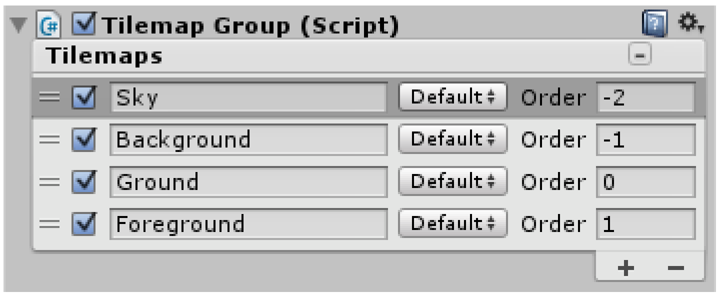
This section shows the collider settings:

- **Collider Type**: None, 2D and 3D
- **Collider Depth**: the depth of the collider (only 3D mode)
- **Collider 2D Type**: the type of collider used: Edge or Polygon (only 2D mode)
- **Is Trigger**: set isTrigger value for all colliders.
- **Update Collider Mesh**: updated the collider mesh. Use this button to update the mesh collider after changing collider settings like Collider Depth.

3.4 Tilemap Groups

A tilemap group is a script used to manage a group of tilemaps like if they were layers.

To create a tilemap group go to "**GameObject->SuperTilemapEditor->TilemapGroup**" or add the TilemapGroup script to an object with tilemap children.



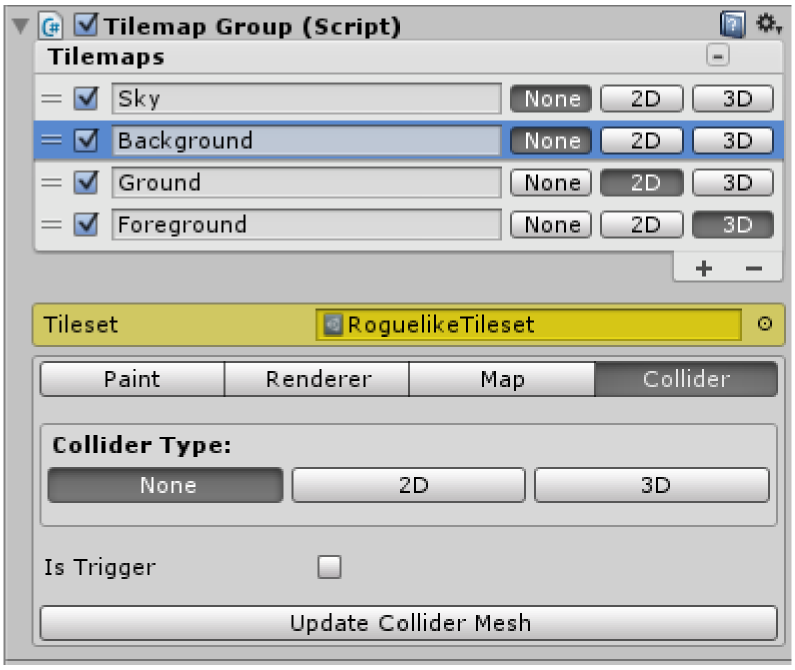
✓ Sky	Default	Order	-2
✓ Background	Default	Order	-1
✓ Ground	Default	Order	0
✓ Foreground	Default	Order	1

In a tilemap group, you can create, remove or select a tilemap directly in the reorderable list.

When a tilemap is selected the tilemap inspector is displayed below.

You can directly access some properties of the tilemap directly from the tilemap list, like sorting layer and sorting order.

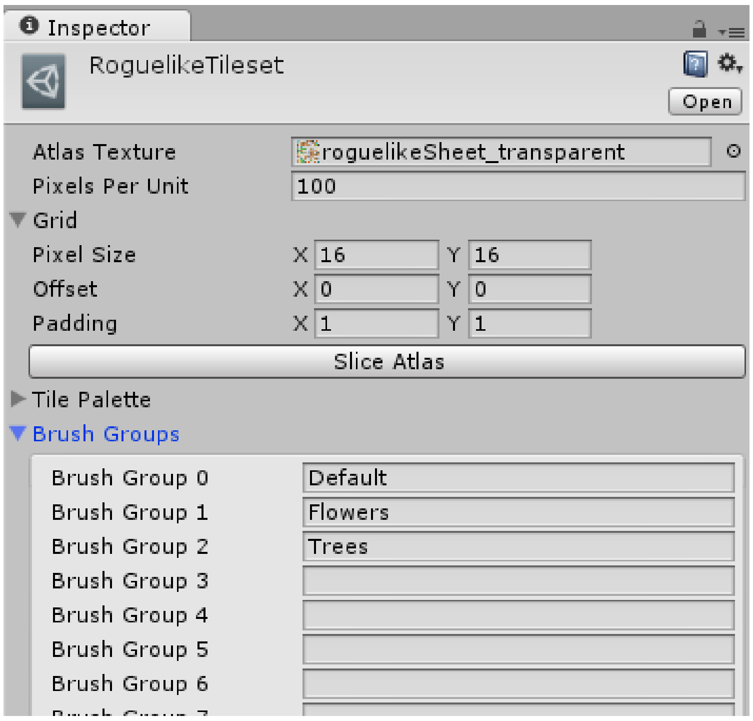
If you select the mode **Collider** you can change also the type of collider for the tilemap.



3.5 *Brush Groups*

A brush group is like a Unity layer used for a game object. You can specify what group a brush belongs to and set the autotiling behaviour for each group.

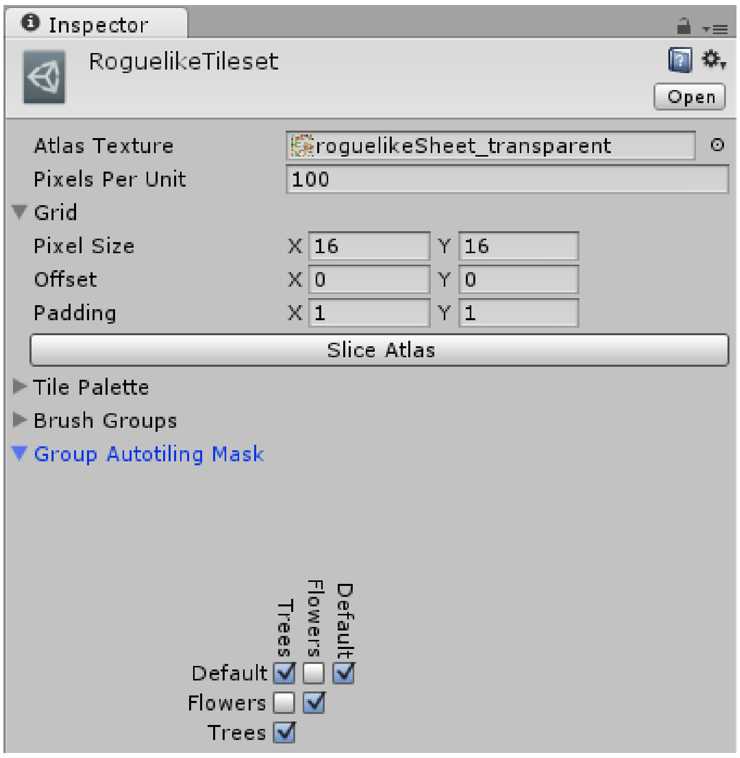
To create a new group, select the tileset and open the **Brush Groups** foldout.


Inspector
RogueliketTileset
Open
Atlas Texture: roguelikeSheet_transparent
Pixels Per Unit: 100
Grid
Pixel Size: X 16 Y 16
Offset: X 0 Y 0
Padding: X 1 Y 1
Slice Atlas
Tile Palette
Brush Groups

Brush Group 0	Default
Brush Group 1	Flowers
Brush Group 2	Trees
Brush Group 3	
Brush Group 4	
Brush Group 5	
Brush Group 6	

You can add a group by setting a name. There are up to 32 groups to be used.

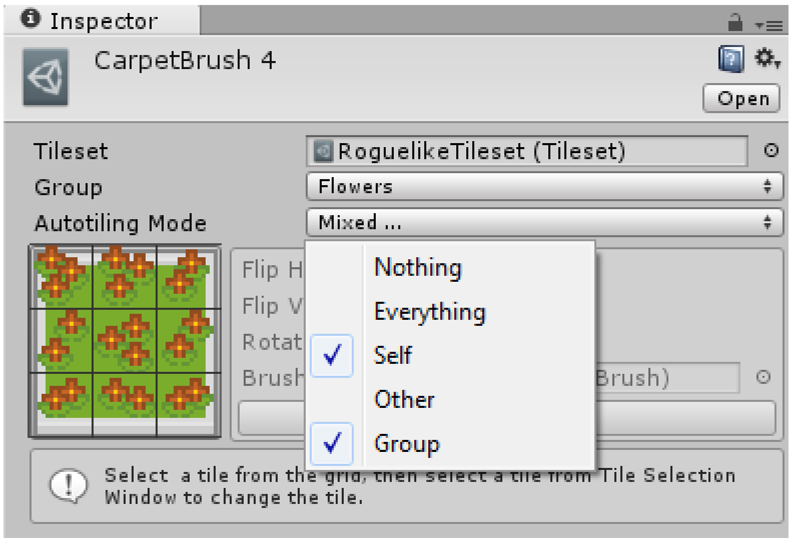
To set the autotiling mask (what groups autotile with what groups), open the **Group Autotiling Mask** foldout.


Inspector
RogueliketTileset
Open
Atlas Texture: roguelikeSheet_transparent
Pixels Per Unit: 100
Grid
Pixel Size: X 16 Y 16
Offset: X 0 Y 0
Padding: X 1 Y 1
Slice Atlas
Tile Palette
Brush Groups
Group Autotiling Mask

	Trees	Flowers	Default
Default	✓	☐	✓
Flowers	☐	✓	
Trees	✓		

You can check when two groups should autotile together by pressing the check box.

All brushes have a **Group** property and an **Autotile Mode.**


Inspector

CarpetBrush 4

Open

Tileset RoguelikeTileset (Tileset)

Group Flowers

Autotiling Mode Mixed ...

Select a tile from the grid, then select a tile from Tile Selection Window to change the tile.

Only a group can be assigned to a brush, but you can mix the Autotiling Modes:

- **Self**: the brush autotiles only with tiles of the same brush
- **Other**: the brush autotiles with tiles of different brush or any tile not empty
- **Group**: checks the **Group Autotiling Mask** to see if the relation between this brush and the neighbor brush is checked to do the autotiling. A normal tile is considered to be in the Default Group (0).

3.6 *Brushes*

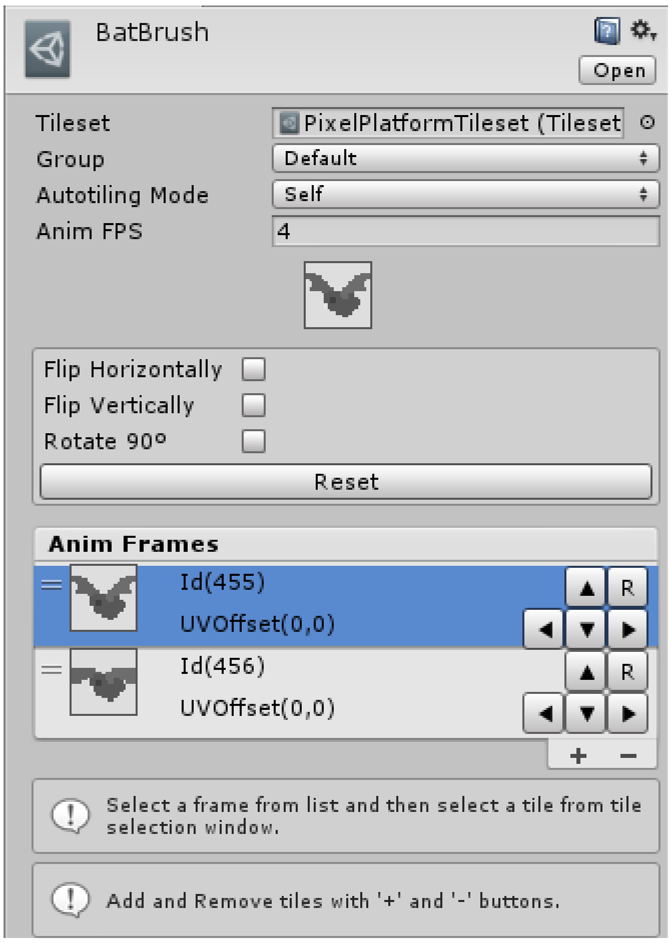
Brushes are scriptable objects derived from **TilesetBrush** and implementing the interface **IBrush**.

All have a common attribute Tileset, so they are associated to an specific tileset.

3.6.1 AnimBrush

To create an animated brush go to:

"**Assets/Create/SuperTilemapEditor/Brush/AnimBrush**"


BatBrush

Tileset: PixelPlatformTileset (Tileset
Group: Default
Autotiling Mode: Self
Anim FPS: 4

Flip Horizontally
Flip Vertically
Rotate 90º

Anim Frames
Id(455) UVOffset(0,0)
Id(456) UVOffset(0,0)

Select a frame from list and then select a tile from tile selection window.

Add and Remove tiles with '+' and '-' buttons.

First set the tileset used by the brush.

Anim FPS will be the frames per second of the animation.

To add a frame to the animation, press the '**+**' button, and press '**-**' button to remove the selected frame.

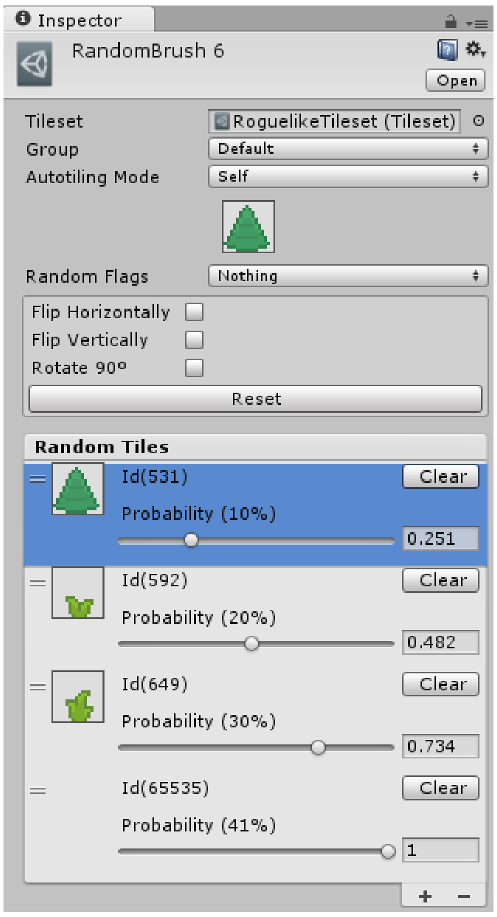
To change the frame tile, select the frame from the list and then select a tile from the tile palette.

You can also add an offset to the frame by pressing the arrow keys or reset the offset by pressing '**R**'.

3.6.2 RandomBrush

To create an animated brush go to:

"**Assets/Create/SuperTilemapEditor/Brush/RandomBrush**"


Inspector
RandomBrush 6
Open
Tileset: RoguelikeTileset (Tileset)
Group: Default
Autotiling Mode: Self
Random Flags: Nothing
Flip Horizontally ☐
Flip Vertically ☐
Rotate 90º ☐
Reset
Random Tiles
Id(531) Clear
Probability (10%) 0.251
Id(592) Clear
Probability (20%) 0.482
Id(649) Clear
Probability (30%) 0.734
Id(65535) Clear
Probability (41%) 1
+ −

First set the tileset used by the brush.

Add or remove tiles from the random list by pressing '+' and '-' buttons.

The first tile will be the preview used in the brush palette.

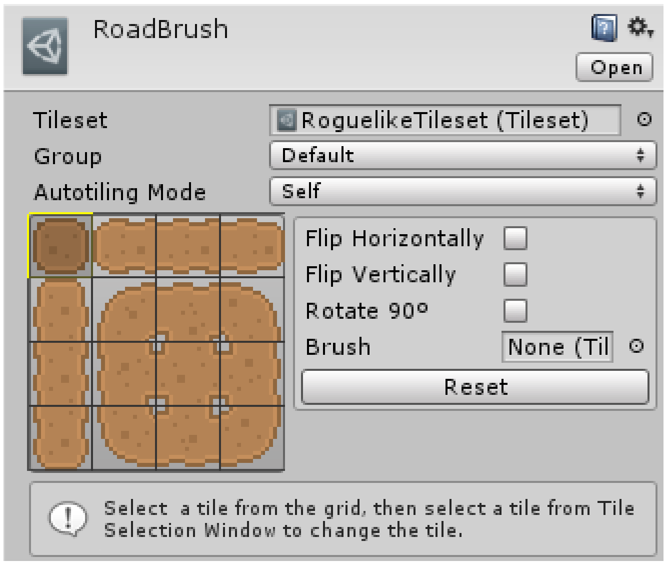
To change the tile, select it and then select another tile from the tile palette.

Random Flags will be the flags that will be randomized when painting the tile, not taking into account the flags set for the tile that are checked.

You can also move the probability weight slide that goes from 0 to 1 to affect the probability each tile have to be chosen.

Also, you can use an empty tile (press Clear button to make it empty) as well.

3.6.3 RoadBrush



RoadBrush

Open

Tileset: RoguelikeTileset (Tileset)

Group: Default

Autotiling Mode: Self

Flip Horizontally ☐

Flip Vertically ☐

Rotate 90º ☐

Brush: None (Til

Reset

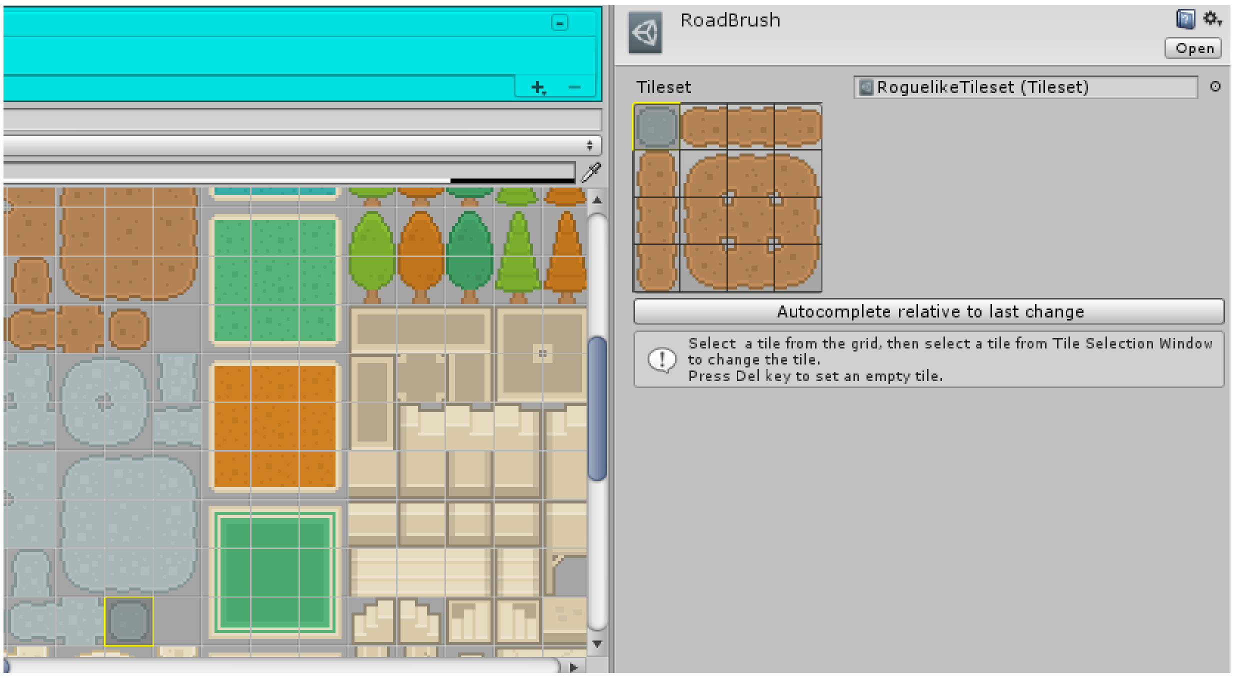
Select a tile from the grid, then select a tile from Tile Selection Window to change the tile.

First set the tileset used by the brush.

This brush is using 16 tiles for each road combination.

Follow the pattern and select one by one the tile from the brush and then select a tile from the tile palette to change it.

If road tiles are placed in same order in the tile atlas, you can create a new road brush faster by changing only one of the tiles of a complete road brush and the pressing the "Autocomplete" button.



RoadBrush

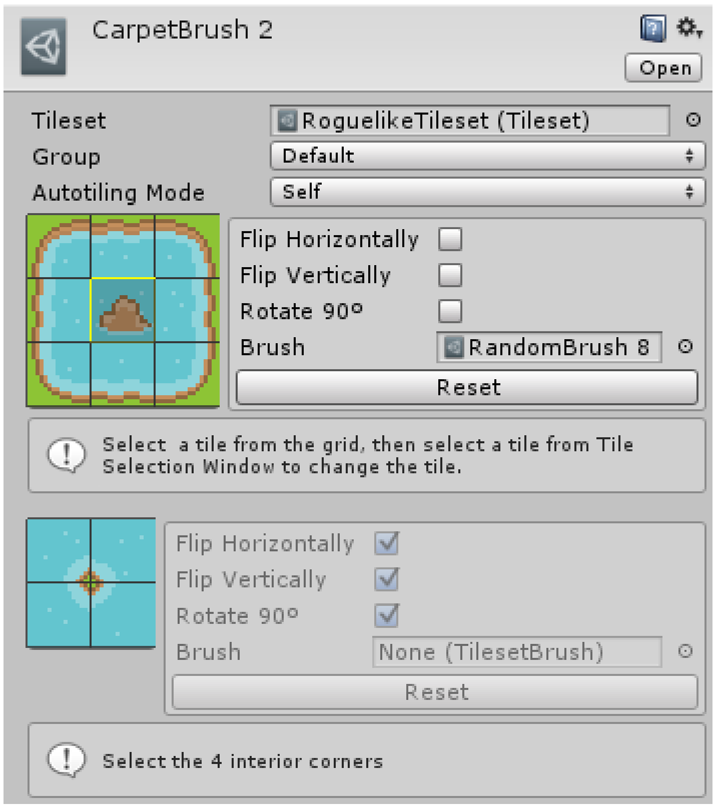
Open

Tileset: RoguelikeTileset (Tileset)

Autocomplete relative to last change

Select a tile from the grid, then select a tile from Tile Selection Window to change the tile.
Press Del key to set an empty tile.

3.6.4 CarpetBrush


CarpetBrush 2
Open
Tileset RoguelikeTileset (Tileset)
Group Default
Autotiling Mode Self
Flip Horizontally
Flip Vertically
Rotate 90º
Brush RandomBrush 8
Reset
Select a tile from the grid, then select a tile from Tile Selection Window to change the tile.
Flip Horizontally ✓
Flip Vertically ✓
Rotate 90º ✓
Brush None (TilesetBrush)
Reset
Select the 4 interior corners

First set the tileset used by the brush.

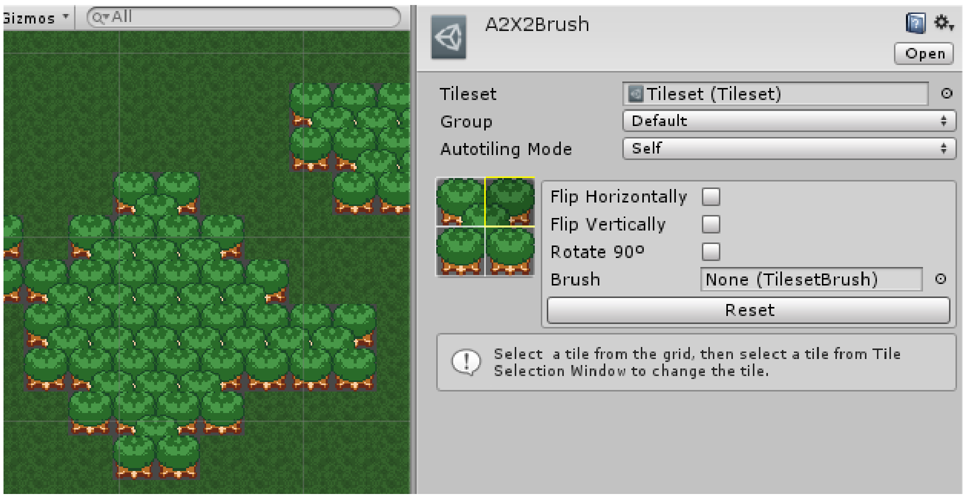
This brush is made to fill areas with interior corners.

Follow the pattern and select one by one the tile from the brush and then select a tile from the tile palette to change it.

If road tiles are placed in same order in the tile atlas, you can create a new road brush faster by changing only one of the tiles of a complete road brush and the pressing the "Autocomplete" button.

You have to autocomplete exterior and interior tiles separately.

3.6.5 A2x2Brush



A2X2Brush

Tileset: Tileset (Tileset)
Group: Default
Autotiling Mode: Self
Flip Horizontally
Flip Vertically
Rotate 90º
Brush: None (TilesetBrush)

Select a tile from the grid, then select a tile from Tile Selection Window to change the tile.

First set the tileset used by the brush.

This brush is made of 4 tiles, one for each corner.

Follow the pattern and select one by one the tile from the brush and then select a tile from the tile palette to change it.

3.7 Custom Brushes

To create a new brush, you have to create a new class derived from **TilesetBrush** and implement the **IBrush** interface.

3.7.1 IBrush Interface

- **uint PreviewTileData()**: the tile data used to show the tile preview.
- **bool IsAnimated()**: return if the tile shold be updated for animation.
- **Rect GetAnimUV()**: return the tile UV for the current frame.
- **uint GetAnimTileData():** return the tile data of the current frame.
- **uint Refresh(Tilemap tilemap, int gridX, int gridY, uint tileData)**: this is called by the tilemap chunks when a tile needs to be refreshed. Resturn the updated tile data.
- **uint[] GetSubtiles(Tilemap tilemap, int gridX, int gridY, uint tileData)**: if a tile is divided in 4 subtiles, this is returning an array of 4 tile data, one per each subtile, from bottom to top, from left to right. Otherwise, it should return null.

3.7.2 TileData

The tile data contains the information used to draw a tilemap tile.

A tile data is stored in a single unsigned int:

- 16-bits[0-15] are used to store the **tileId**
- 12-bits[16-27] are used to store the **brushId** used to draw the tile
- 4bits[28,29,30,31] are used: 28 tile **updated**; 29 for **90 degrees rotation**; 30 for **vertical flip** and 31 for **horizontal flip**.

If you need to know the rotation degrees of a tile, you have to check the value [0, 1] of the flags Rot90 (90 degrees rotation), FlipV (flip vertical) and flipH (flip horizontal):

Rotation	**Flip Vertical**	**Flip Horizontal**	**Rot90**
0º	0	0	0
90º	0	0	1
180º	1	1	0
270º	1	1	1

To manage uint values of tiledata you need to know how to works with bitwise operators. You can find in Tileset class all the constants you need for setting and getting data from tile data, but to make it easier, there is a class TileData you can use to deal with tile data.

Let's see some examples of using this class:

- **Read tile data from tilemap position (12, 45):**

  uint rawTileData = tilemap.GetTileData(12, 45);

  TileData data = new TileData(rawTileData);

  bool flipVertical = data.flipVertical;

  bool flipHorizontal = data.flipHorizontal ;

  bool rot90 = data.rot90 ;

  bool brushId = data.brushId ;

  bool tileId = data. tileId ;

- **Write tile data at tilemap position (50, 13)**

  TileData data = new TileData();

  //Fill all data you need in data (remember a data without brushId or tileId is considered as an empty tile)

  tilemap.SetTileData(12, 45, data.BuildData());

4 Toolbar Menu

In the toolbar menu, you can find the SuperTilemapEditor with more options:

- **Brush**
  - **Create Tilemap From Selection**: will create a tilemap using the current brush selection
  - **Create Prefab From Selection**: will create a prefab using the current brush selection
- **Window**
  - **Tile Properties Window**: will display the tile properties window
  - **Tile Palette Window**: will display the tile palette window

5 Importing from TMX files

You can create a tileset from a tmx files and once you have the tileset, you can import the tmx scene into the unity scene create a TilemapGroup in the process.

To do that you have to follow the next steps.

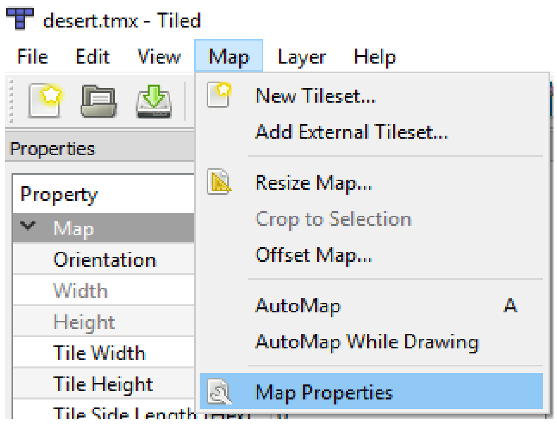
5.1 Preparing the TMX file before importing

To import a tmx file you need to save the tmx in XML format first.

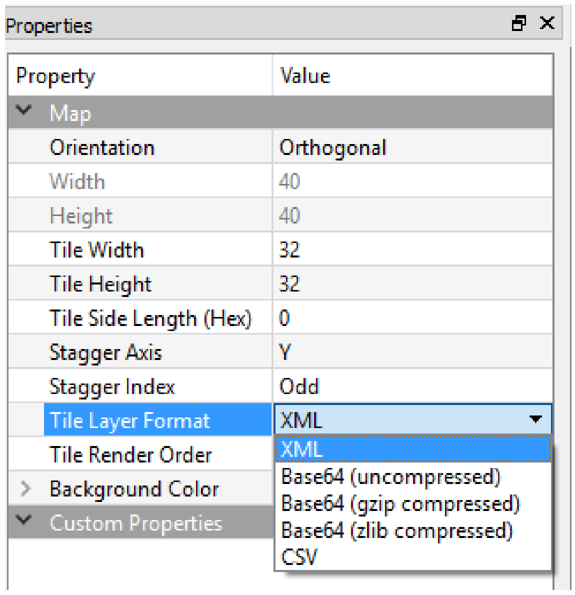
Open the tmx file with Tiled.

Select the option Map Properties from the menu.

"Map->Map Propertiies"



Select the XML option for the Tile Layer Format property.



Property	Value
Map	
Orientation	Orthogonal
Width	40
Height	40
Tile Width	32
Tile Height	32
Tile Side Length (Hex)	0
Stagger Axis	Y
Stagger Index	Odd
Tile Layer Format	XML
Tile Render Order	
Background Color	
Custom Properties	

Save the file.

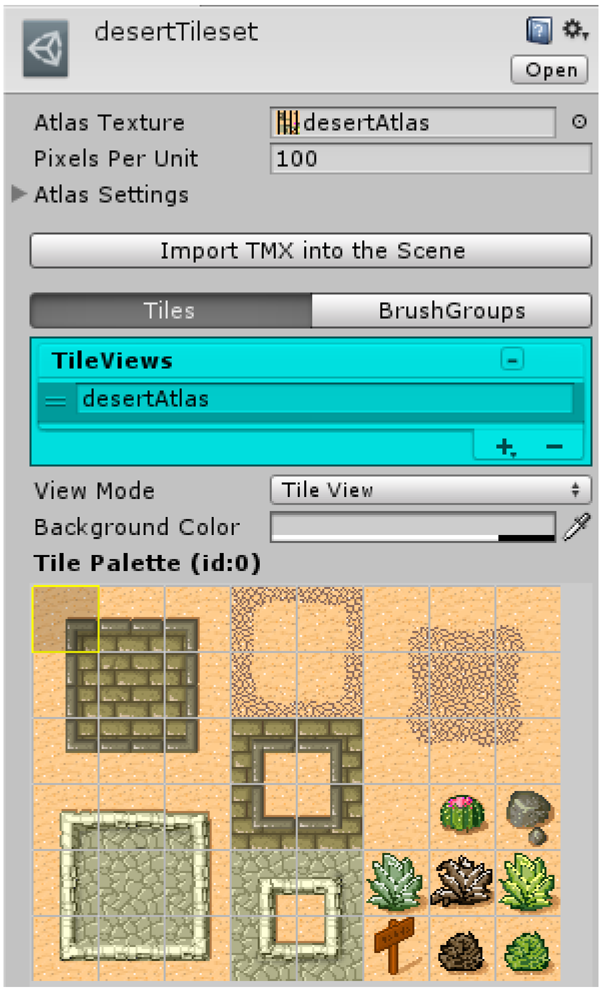
5.2 *Creating a tileset from TMX file*

Go to "**Assets/Create/SuperTilemapEditor/Tiled/Tileset (from TMX File)**".

Select the tmx file to create a compatible tileset.

A tileview will be created for each tilemap found in the tmx file, and an atlas png file will be created as well with all images used by the tilesets.

A tileview will be also created for each tileset.



desertTileset

Atlas Texture: desertAtlas
Pixels Per Unit: 100
Atlas Settings

View Mode: Tile View
Background Color
Tile Palette (id:0)

5.3 *Importing a tmx file into the scene*

Select the tileset created in the previous step. Press the button "**Import TMX into the scene**".

Select the tmx file to create a tilemap group into the scene with a tilemap for each layer found in the tmx file.

6 Advanced Topics

In this section I will explain in deep how things work in Super Tilemap Editor and other useful information.

6.1 The Tilemap and tile rendering

A tilemap is a game object with contains other children objects called tilemapChunks (and they are hidden by default but can be showed by checking Show Map Tilechunks in the Map section of a tilemap).

The reason on this is, to render the tiles, a mesh is created with all tiles contained in the tilemap, but there is some limitations in how many vertices a mesh can have of 65535.

Because of this limitation, the tilemap is split in tilemapChunks and this children object will contain the mesh to draw a section of the tilemap.

These chunks are created only when needed, this is, when there is at least a tile to be rendered.

There is constant in the Tilemap class, **k_chunkSize**. This constant is the size of each chunk, so the tilemap will be subdivided by this size.

TIP: You can see the chunks in the tilemap surrounded by red lines.

You can modify this constant, but all the tilemaps created with a different size will be invalidated.

If you change this value, you need to know about the mesh vertex limitation.

Each tile will need 4 vertices, so if you do the math, the maximum size would be 127 (4x127x127 = 64516).

If you have into account a tile can be subdivided in 4 subtiles (ex: using a Carpet Brush), then you have to divide by 4.

So a safe value to make sure you never will overflow the limit would be 63 (I set the value to 60).

TIP: Each tilemapChunk being rendered by the camera (not culled) will count as a single draw call.

6.2 *Updating a Tilemap by scripting*

You can modify the tilemap by calling **SetTileData**. There are multiple overloaded methods but they all do the same, change the tile data information in a tilemap grid position.

The tilemap is divided in a grid with cells and each cell contains the information to draw a tile or a brush and also flags like horizontal and vertical flip and 90 degrees rotation.

This information is stored in a single unsigned int of 32bits for performance.

Check section 3.7.2. TileData for more information about how the data is stored.

To set a grid position as empty you should call **Erase**.

But calling **SetTileData** or **Erase** won't update the tilemap mesh renderer neither the colliders until you call **UpdateMesh** or **UpdateMeshImmediate**.

6.3 *TileObjectBehaviour and OnTilePrefabCreation event*

When a prefab is selected in the Prefab section of the properties window, the prefab will be instantiated where the tile is painted.

When this prefab is instantiated, a message will be send to the instantiated game object. To catch this message you should declare this method in any of the attached components of the game object:

void OnTilePrefabCreation(**TilemapChunk.OnTilePrefabCreationData** data).

OnTilePrefabCreationData contains some useful information:

- **Tilemap ParentTilemap**: the tilemap creating this game object
- **int GridX**: the grid X position where this object was created
- **int GridY**: the grid Y position where this object was created

You can use this information to get the properties of the tile used to create this object, for example.

There is a script component included in the project called **TileObjectBehaviour.**

When this object is attached to a prefab, it is using the OnTilePrefabCreation message to get create a sprite with the tile uv and set this sprite in the SpriteRenderer of the game object:

```
public class TileObjectBehaviour : MonoBehaviour
{
    void OnTilePrefabCreation(TilemapChunk.OnTilePrefabCreationData data)
    {
        Tile tile = data.ParentTilemap.GetTile(data.GridX, data.GridY);
        if (tile != null)
        {
            float pixelsPerUnit = data.ParentTilemap.Tileset.TilePxSize.x /
data.ParentTilemap.CellSize.x;
            Vector2 atlasSize = new Vector2(data.ParentTilemap.Tileset.AtlasTexture.width,
data.ParentTilemap.Tileset.AtlasTexture.height);
            Rect spriteUV = new Rect( Vector2.Scale(tile.uv.position, atlasSize),
Vector2.Scale(tile.uv.size, atlasSize));
            SpriteRenderer spriteRenderer = GetComponent<SpriteRenderer>();
            spriteRenderer.sprite = Sprite.Create(data.ParentTilemap.Tileset.AtlasTexture, spriteUV,
new Vector2(.5f, .5f), pixelsPerUnit);
            spriteRenderer.sortingLayerID = data.ParentTilemap.SortingLayerID;
            spriteRenderer.sortingOrder = data.ParentTilemap.OrderInLayer;
        }
    }
}
```

7 F.A.Q

I see weird lines around the tiles when I am painting a tilemap?

Check if the atlas texture used by the tileset have the filter mode diffent than None in the texture import settings.

How can I remove a tile from a tilemap position?

tilemap.Erase(gridPositionX, gridPositionY);

I called SetTileData and/or Erase but the tilemap is not updated

Call tilemap.UpdateMesh().

How can I get the center position of a tile in a tilemap?

Call TilemapUtils.GetGridWorldPos

How can I get the tilemap grid position where the mouse is over?

Call TilemapUtils.GetMouseGridX and TilemapUtils.GetMouseGridY

How can I create a tile as a separated game object?

Create a prefab of a game object with the component TileObjectBehaviour attached. Attach this prefab to the tile using the tile property window, section Prefab. This prefab will be instantiated where the tile is located looking like the tile but being a separated game object.

How can I access the collided tilemap when receiving a OnCollision event?

The tilemap is a parent object with invisible children objects containing the tilemap chunks. The object you collided with will be one of these tileChunks so to access the tilemap take the parent of the collided tilechunk object.

```
void OnCollisionEnter(Collision other){
Tilemap tilemap = other.gameObject.GetComponentInParent<Tilemap>();
}
```

The player trespass the wall or the 2D colliders doesn't connect properly

Make sure you are using Edge Colliders and IsTrigger is not checked. Polygon Colliders or colliders with isTrigger set to true cannot be concave and should be used carefully.

Also, make sure the normal of the collider is facing the right side of the collider lines. In most cases they should be facing from inside to the outside. Press the Reverse Normals button in the tile collider properties window if needed.