

# **YuME FREE: Yuponic Map Editor**

Version: 1.0  
Initial release

# **1. YuME: Yuponc Map Editor**

---

## **1.1. Introduction**

---

The Yuponc Map Editor (YuME) is a Unity Editor Extension created to allow level designer to rapidly prototype game levels and environments using 3D Tiles, or blocks.

The full version of YuME allows you to create your own tilesets and import them in an intuitive way for use in the tool and your game. You can also convert existing tile-based assets for use in the tool.

### **1.1.1. Why did we create this tool?**

---

Unity is a fantastic game engine that allows amateurs and pro's alike to create the game of their dreams. However, in catering for all possible game types, from MMOs to Flappy Birds, the editor needs to function in a genre neutral way.

Building game levels in Unity can be time consuming and fiddly due to limited grid snapping and the time consuming placement of prefabs.

YuME allows designers to build levels in a more natural way by using a set of prefab brushes and painting them directly into a grid snapped scene.

This approach will be very familiar to 2D game creators, and with YuME I wanted to bring the speed and flexibility of a 2D Tile Mapper to 3D games.

### **1.1.2. A little history**

---

3D Tiles form a large part of my history in the games industry. I've been fortunate enough to have making games professionally for over 23 years, and throughout that time the techniques and tools for game creation have changed dramatically, but in other ways have also come full circle.

I first worked with a 3D tile based editor on the game Vortex for the Super Nintendo. Back then the maps were built in D-Paint (an industry standard bitmap editor). We had a limited palette which represented game objects and dropping a color on a 64x64 bitmap to place an object in the scene.

A few years later I started working on a concept for a 3D Platform Game that became Croc: Legend of the Gobbos. Realizing we needed a tool to build the levels we ended up creating CrocEd, which used 3D tiles that were placed within a cube grid to build the world.

Although the CrocEd was ultimately limited by what we could create on the Playstation, the tool allowed the design team to be incredibly creative and iterate quickly on levels for the game.

The CrocEd and engine went on to power a bunch of different games, from Disney's Aladdin to Harry Potter.

As the trend to realism kicked in throughout the PS2/PS3 years, tile based games fell out of favor. That is until recently.

As the market broadened I started to see a lot more games go back to using techniques we developed for Croc. Even Nintendo moved over to a more tile-based approach with games like Super Mario 3D World and Captain Toad Treasure Tracker.

These days it's an artistic choice rather than a limitation of technology. And for me, it's a fast and creative way to try out ideas without having to build geometry and hand place it in the scene.

### 1.1.3. Suitability

---

YuME is the perfect tool for creating focused, level based games. These can be strategy games, platform games, arena based FPS games, etc.

It's probably better to focus on games YuME isn't so suited to.

If you're looking to create a vast open world game with a naturalistic environment, be it a mystic forest or the streets a cyber-Singapore then YuME isn't going to get you to your end goal. There are tools and techniques better suited to creating those kind of games, and I probably owe you a refund 😊

## 1.2. Use Cases

---

There are a few use cases where I can see YuME being the most useful:

- **Tile Based Games:**  
Where game environments are designed around the concept of tile prefabs. Suggested games to look at include Captain Toad Treasure Tracker, Atlas Reactor, and the latest batch of Pokémon games for the 3DS.
- **Level Prototypes:**  
Where a level designer wants to quickly build out the structure of the level with an eye to an art team taking the basic structure and doing an art pass.
- **Game Jams and Prototypes:**  
Where you quickly want to try out ideas and mechanics and make rapid progress.

For some games, YuME will be the only tool you use to build levels. For others it will be the start of the tool chain to help you get up and running quickly and efficiently.

## 1.3. Feature requests and Bugs

---

We're always looking for suggests on how to improve YuME, and we're also keen to squash any bug you discover. If you have any suggestions or bugs to report, either drop by the facebook page or drop me an email on [info@yuponic.com](mailto:info@yuponic.com)

## 2. Getting Started

---

### 2.1. Importing YuME

---

Once purchased from the Unity Asset Store you'll be prompted to import two key folders, one name Yuponc and the other Gizmos.

The Yuponc folder is the home of YuME and the source for the editor and prototype tileset. The Gizmos folder contains a single image which is used as a guide to where tiles are located in the scene.

YuME has been designed so that folders are not hard-coded. This means that after import you can move the Yuponc folder into a sub-folder of you wish. For example, I like to move all of my 3<sup>rd</sup> party assets into a specific folder to keep the root of the project tidy.

### 2.2. Opening the Map Editor

---

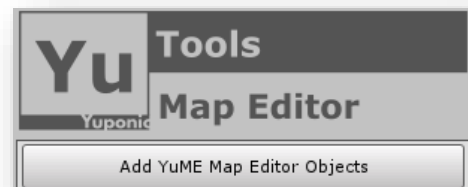
Once imported the window drop down will have a new entry named Yuponc. Instead the Yuponc drop down you'll see three items and a sub-folder. Selecting 'YuME: Map Editor' opens up the map editor.

The map editor window can be docked or left floating if based on your preference.

### 2.3. Adding YuME to a Scene

---

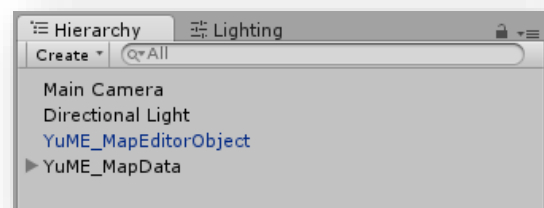
YuME needs to add a couple of game objects to your scene. One object handles the grid and is not active in play mode. The other acts as a container for the tiles you'll be dropping into the scene.



To add these objects to the scene, click on the 'Add YuME Map Editor Objects' button in the map editor window.

The scene will now have two new objects:

- YuME\_MapEditorObject
- YuME\_MapData



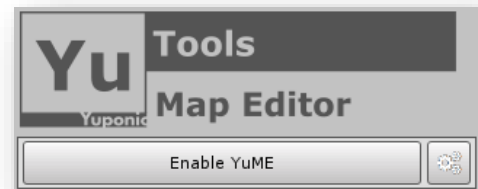
## 2.4. Enabling YuME

---

Once the scene and the necessary files are added to the scene, the main map editor interface is displayed.

To start using YuME, click the 'Enable YuME'.

The YuME grid and tool bars will appear in the scene view and you are ready to start building your level.

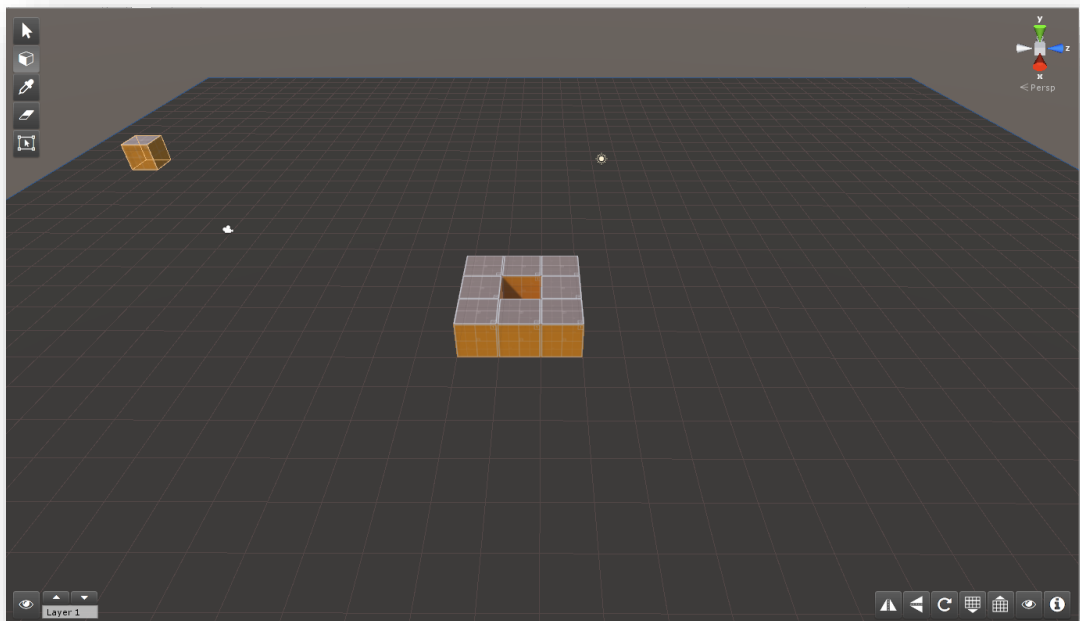


## 2.5. Building Levels

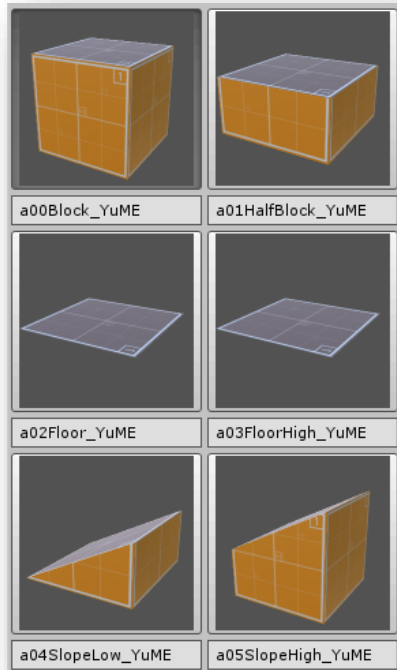
---

By default, the Paint Tool (3D cube icon) is active. Using this tool, you can paint the currently selected tile onto the grid.

If this is the first time you've used YuME, the prototype tileset will be set as the current tileset.



As you move your cursor over the scene, you'll see a highlighted copy of the currently selected tile move across the grid. To begin painting tiles in the scene, simply press or hold the left mouse button.



To change the tile that forms the brush simply select a tile from the palette in the map editor window.

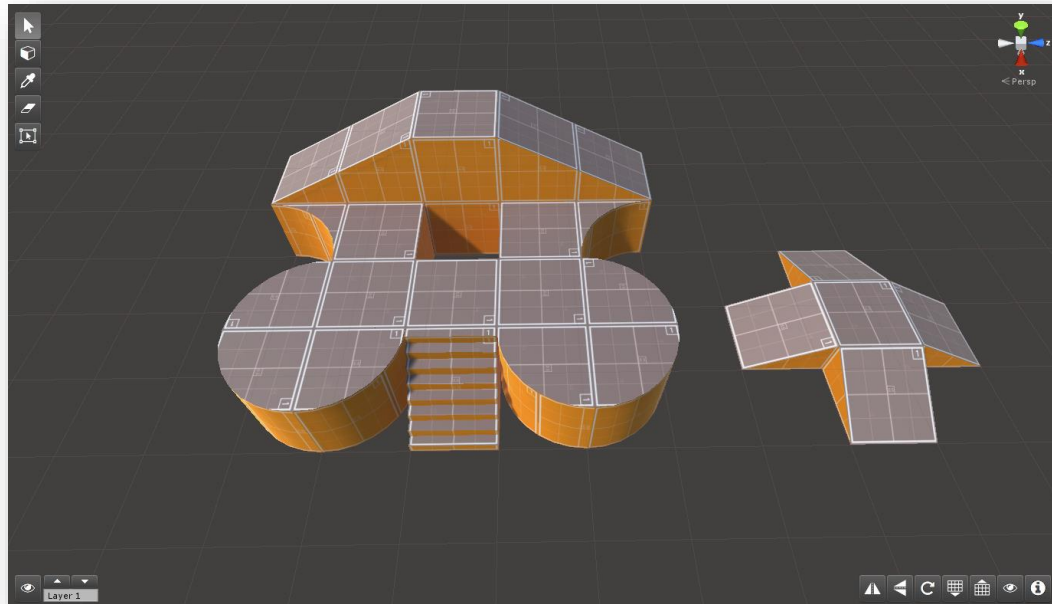
The palette window is a great way to preview the content you can build your levels with. However, if you want to quickly scroll through the available tiles, hold SHIFT and CONTROL and scroll the Mouse Wheel.

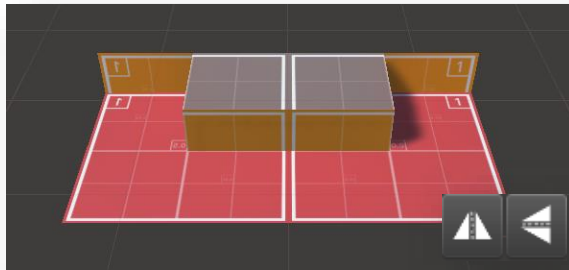
With every scroll of the mouse wheel the brush cursor updates allowing you to stay focused on the scene view as you create your levels.

To build on different levels you can move the grid up and down. This can be done by either clicking on the grid icons or by holding SHIFT and scrolling the Mouse Wheel.



Often you'll want to rotate tiles that are none symmetrical. To rotate the tile, you can either click on the rotate icon or use the Z and X key to rotate clockwise and counter-clockwise.

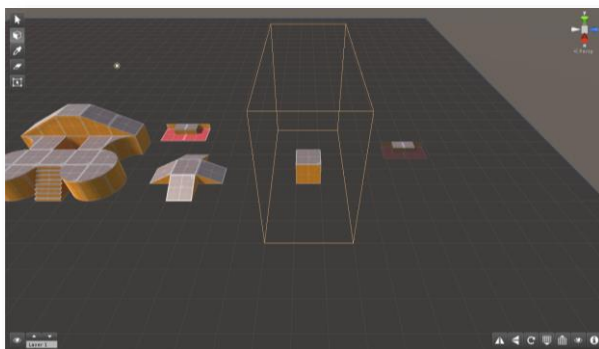
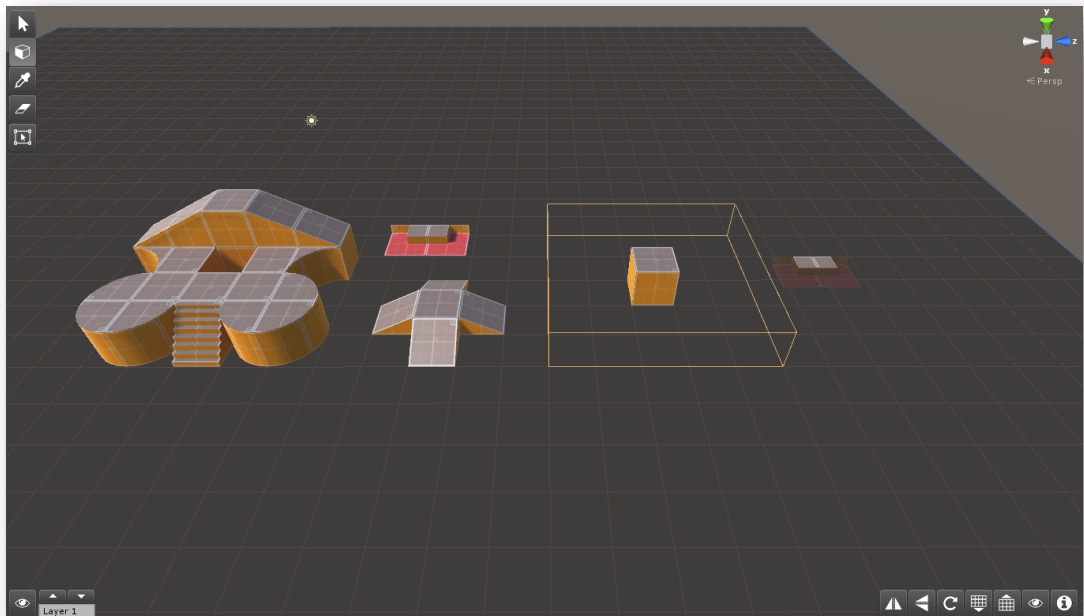




Tiles can also be flipped, allowing for the maximum reuse of the minimum amount of tiles.

To flip a tile horizontally or vertically simply click on the flip icons.

If you want to paint more than one tile at a time you can adjust the brush size. There are a couple of different ways to do this. To simply 2D scale up the brush, use the [ and ] keys.



You can also independently scale the brush on all axis.

Cursor Left/Right scales the brush on the X axis.

Cursor Up/Down scales the brush on the Z axis. Shift and cursor Up/Down scales the brush on the Y axis.

Scaling on the X and Y always scales to odd numbers (for example 1,3,5,7) to keep the brush centered on the grid. Scaling on the Y is purely incremental.






To reset the brush press Enter/Return.

## 3. Tools Overview

---








### 3.1. Primary Tool Bar

---

	<b>Standard Tools:</b> Defaults back to the standard unity tools. Tool shortcut: ESC
	<b>Paint Tool:</b> Paint Tiles into the scene. LMB to Paint the current brush. Tool shortcut: A
	<b>Pick Tool:</b> Pick a tile in the scene and make it the current brush. LMB to pick tile. Tool shortcut: S
	<b>Erase Tool:</b> Erase the tile under the brush. LMB to erase the tile. Tool shortcut: D
	<b>Select Tool:</b> Select the tile under the brush. LMB to select the tile. ALT + LMB to deselect.

### 3.2. Secondary Tool Bar

---





	<b>Info Tool:</b> Show a helper gizmo that identifies the tile pivot. If tiles are selected, additional information about their layer, name, and grid level is shown.
	<b>Isolate Grid Tool:</b> Hides all tiles not on the current grid level.
	<b>Grid Up Tool:</b> Move the grid up 1 position. ALT + click moves the grid 0.25. Tool shortcut: = or SHIFT + Mouse Wheel Up
	<b>Grid Down Tool:</b> Move the grid down 1 position. ALT + click moves the grid 0.25. Tool shortcut: - or SHIFT + Mouse Wheel Down
	<b>Rotate Tool:</b> Rotate the current brush 90 degrees. Tool shortcut: Z / X to rotate clockwise / counter-clockwise
	<b>Flip Z Tool:</b> Flips the brush in the Z axis.
	<b>Flip X Tool:</b> Flips the brush in the X axis.



### 3.3. Select Toolbar


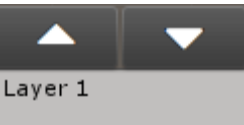
---

The select toolbar is only active when a tile is selected. To select tiles use the Select Tool or the standard unity tools.

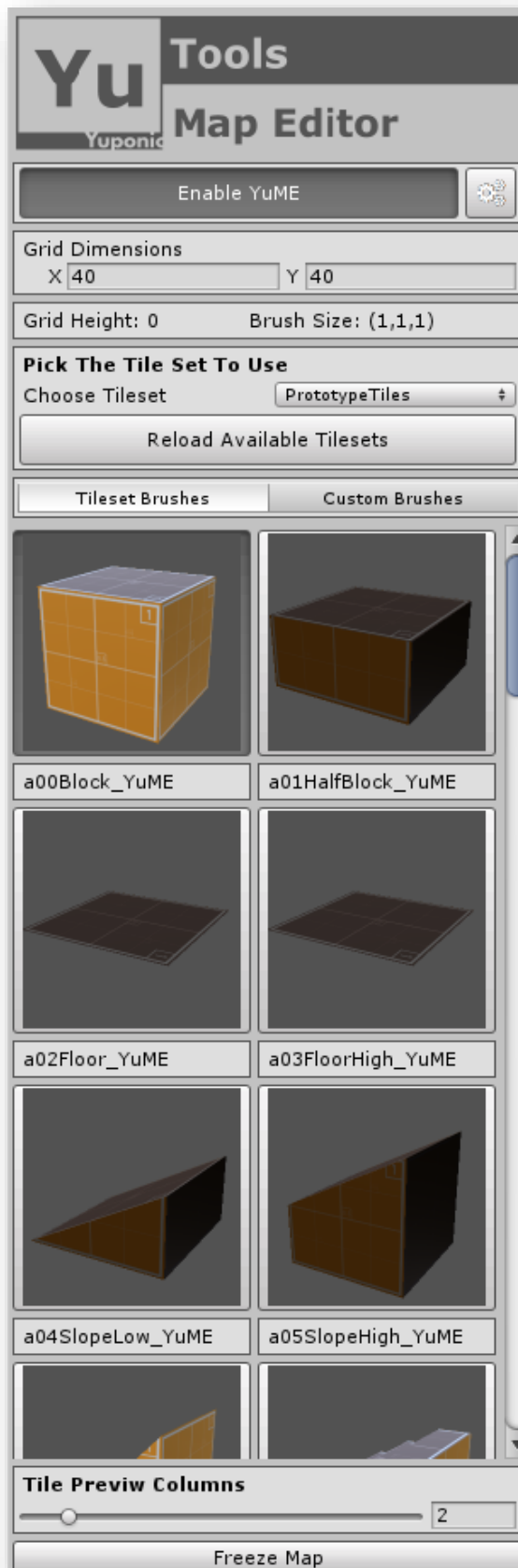
	<b>Copy Brush Tool:</b> Copies the currently selected tiles and makes them the active brush.
	<b>Cut Brush Tool:</b> Cuts the currently selected tiles and makes them the active brush.
	<b>Custom Brush Tool:</b> Creates a saved copy of the selected tile and stores them in the custom brush tab and folder.
	<b>Erase Selected Tiles Tool:</b> Deletes the selected tiles from the scene. Important. When selecting with the default unity tools you should always use the erase tool to prevent additional game objects from being delete.

### 3.4. Layer Toolbar

---

	<b>Layer Isolate Tool:</b> Hide all tiles that are not on the current layer.
	<b>Layer Select Tool:</b> Use the arrows to select the current layer. Note: Layer names can be edited in the settings window.

## 4. Map Editor Window



### Enable YuME:

Activates the editor in scene. In an off state all standard Unity functionality is restored

### Open Settings:

Opens the settings window.

### Grid Dimensions:

Change the size of the YuME grid.

**Scene Info:** Current grid height and brush size.

**Choose Tileset:** Pick the active tileset dropdown.

### Reload Tilesets:

Refreshes the available tileset. Used after import.

### Brush Types:

Choose standard tiles or custom brushes

### Brush Palette:

Previews of all the current tiles in a tileset. Click select to make a tile the active brush.

### Preview Columns:

Change the width of the palette to suit your editor setup.

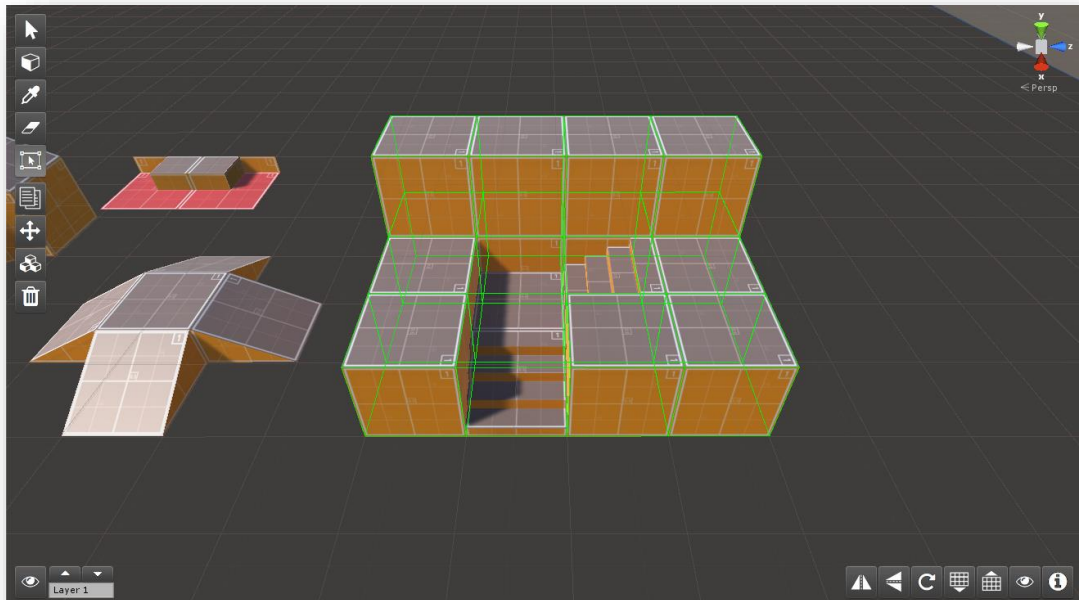
### Freeze Map:

Consolidate the map into a single, un-editable mesh.

## 5. Custom Brushes

### 5.1. Creating a custom brush

Often you'll find yourself repeating specific tile configurations. To save time in level building you can save a selection of tiles and store them as a custom brush.

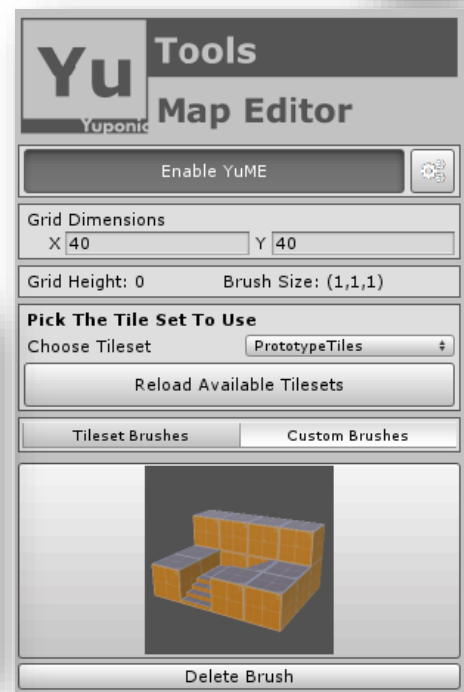
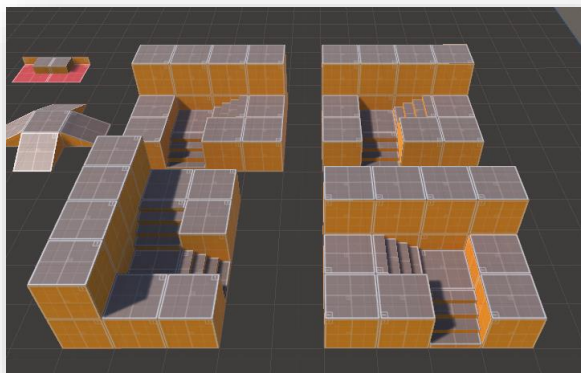


Select the tiles you want to store either using the Select Tool of the standard unity selection tools. Once selected click on the Custom Brush icon.



The custom brush will now be available in the custom brush tab of the main YuME editor window.

To use the custom brush simply select. Custom brushes can be rotated and flipped just like standard tiles.

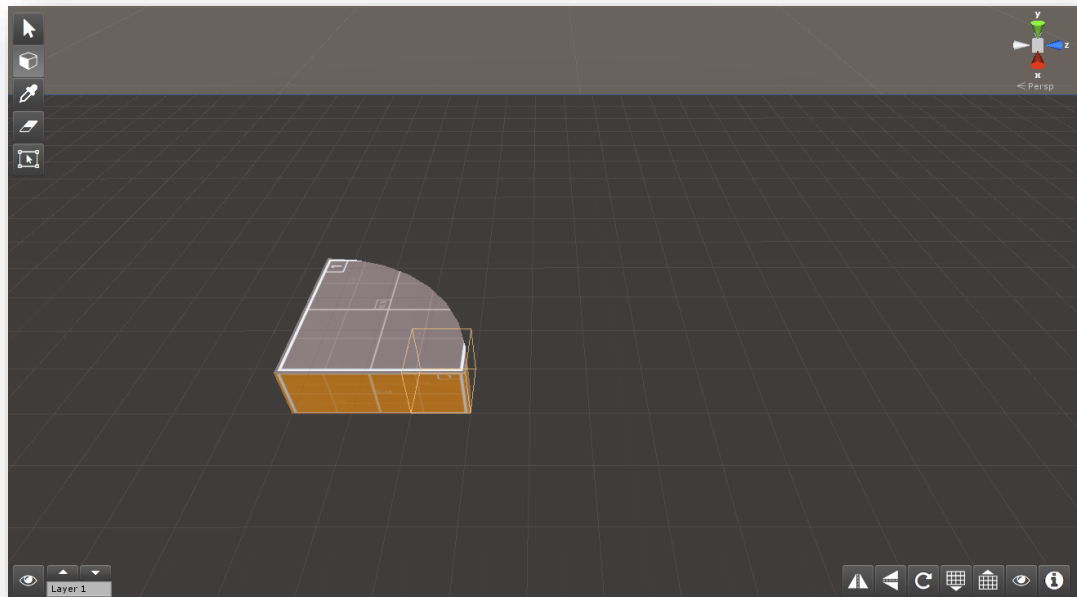


If the brush is no longer needed it can be delete in the editor window.

## 6. Non-standard Tiles

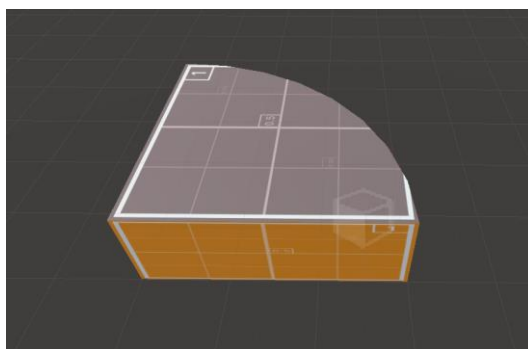
### 6.1. Considerations for non-standard tiles

YuME is designed to work with tiles built around a 1x1x1 cube. However, tiles of different sizes can be used in YuME with some minor caveats.



In the example above we have a 3x3 tile. If you look closely at the cursor you can see it's pivot is the right side corner. If you wish to delete or pick the tile, then this corner position is the only selectable point on the tile.

To make it easier to see where the pivot point on a placed tile is location you can use the information tool.



Turning on the information places a unity gizmo at the pivot point of each tile in the scene.

Where possible it's best practice to build larger blocks as custom brushes. However, there are obviously cases where this is inefficient or impractical.

## 7. YuME Settings

---

### 7.1. Changing the editor settings

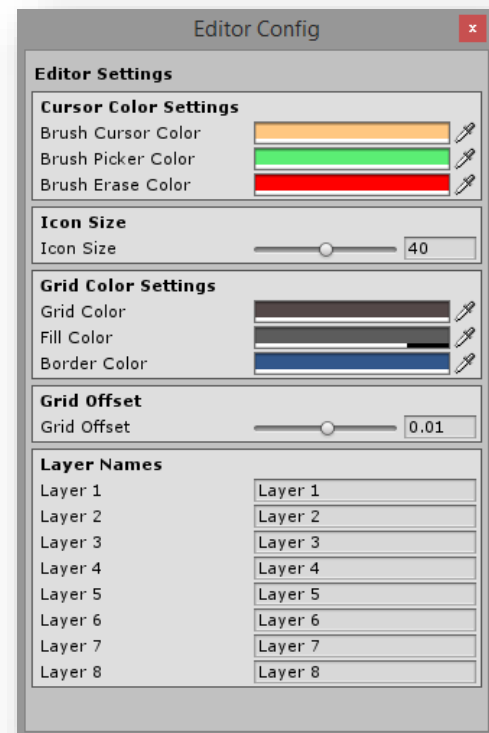
---

Various aspects of YuME can be customized in the Editor Config window. The window can be accessed by clicking on the config button next the 'Enable YuME' button.

Once the editor config is open you can select colors for the brush cursor, the editor icon size, and the grid colors and opacity.

Grid offset allows you to fine tune the offset of the grid. This is useful for certain tilesets that either sit under or over the normal grid position.

Finally, you can rename the layers. This can be very useful for projects where specific tiles need to be dropped on set layers.



## 8. Shortcuts

---

Select Tool:	ESC key
Paint Tool:	A
Pick Tool:	S or Paint Tool + CONTROL
Erase Tool:	D or Paint Tool + SHIFT
Cycle through Tiles:	SHIFT + CONTROL + Mouse Wheel
Grid Up/Down:	= - or SHIFT + Mouse Wheel
Grid Up/Down 0.25:	SHIFT + ALT + Mouse Wheel
Brush Size 2D:	[ ]
Brush Size +X:	Cursor Right
Brush Size -X:	Cursor Left
Brush Size +Z:	Cursor Up
Brush Size -Z:	Cursor Down
Brush Size +Y:	SHIFT + Cursor Up
Brush Size -Y:	SHIFT + Cursor Down
Reset Brush:	Enter/Return