Quick guide to using the MicroSims implementation in Unity.

# Running it

Just click on the play button as with anything else in Unity. Press it again to kill it, or press pause to pause/unpause it. You can edit objects through the inspector at any time during the game and you can edit things through the scene view (i.e. drag objects around) while it’s paused.

# Editing the world

## Selecting and moving objects

This is mostly like other Unity stuff. You can select objects using the Hierarchy panel on the left. Windows users can also usually select objects by clicking on them.[[1]](#footnote-1)

## Moving objects

Characters and objects can be dragged around to place them (again, Windows only). Most objects will snap to the grid of the tile map, but characters can be placed freely. You can also move them by directly editing the Transform component of the objects. Only the X and Y properties are supported; Z, scaling, and rotation are ignored.

# Adding characters

This is probably most easily done by selecting the existing character in the Hierarchy view and choosing Duplicate from the Edit menu. Then rename the new character, move it to a different location, and edit to your hearts’ content.

## Changing the character’s appearance

To change the character’s appearance, select the character in the Hierarchy view, then change the **Texture** field of its **Row Animation Sheet** component to the spritesheet of your choice. There are a number character sheets in the Characters folder of the Assets panel. There are many public domain sources for sprite sheets. These particular ones come from [Sithjester's RXMP Resources](http://untamed.wild-refuge.net/rmxpresources.php?characters), which also has many more. Simply download the spritesheet of your choice (it’s just a PNG file), drag it into the Characters folder, and then drag it from there to the Texture field of your character’s Row Animation Sheet.

# Adding new props

Again, select an existing object, duplicate, rename, and move it. Then change the tile it displays using the **Select tile** popup inside the TileSprite component.

## Changing the object’s footprint

Each TileSprite has a pair of rectangles associated with it, its footprint and its docking rectangle. The footprint specifies what pixels the object “occupies” in the virtual world. It’s the way we prevent the characters from walking through objects: the characters won’t walk through any tile in the object’s footprint.

The object’s footprint is displayed as a **red** rectangle in the scene view when the object is selected in the editor. You basically need to play with the numbers listed under “footprint” in the Inspector until the red rectangle covers the object without going too far beyond it.

## Changing the object’s docking rectangle

This is the area that a character is supposed to stand in when “using” the object. Like the footprint, it’s a colored rectangle that’s displayed in the editor when the object is selected. It’s displayed in **green**. Also like the footprint, it’s controlled by adjusting the numbers in the inspector, only you adjust the numbers under “docking rectangle” rather than “footprint”.

## Changing advertisements

Advertisements tell characters what needs a given object can satisfy. If an object is a prop that is just there to look nice, you should delete any advertisements it has. If the object is there to let the character “use” it to satisfy a need, then it needs an advertisement. If the object you duplicated didn’t have one, then make sure the object has been selected, then go to the Components menu, choose Sims and then “Simple advertisement”. Then edit its “satisfaction” in the inspector to specify the need it satisfies and how many points of satisfaction it gives, on a 0-100 scale.

The other fields under the Simple Advertisement control how the character actually interacts with the object. For Simple Advertisements, the character just walks up to the object and stands next to it for a certain period of time and that constitutes “using” it. You can customize this by editing Duration (how many seconds it stands next to the object), and the start speech, during speech and end speech, each of which are just canned pieces of text it displays while walking to the object, using the object, and when done with the object, respectively.

It’s also possible to have an object satisfy multiple needs at once by adding more elements to the Satisfactions array (select one element, select Duplicate from the Edit menu, and then edit the new element). You can also have it satisfy different needs in different ways by adding two different advertisements, each with a different satisfaction and different duration and speeches.

# Editing the map (Windows only, sorry!)

Editing the map is uglier than I’d like, but it’s the best I could do given my limited understanding of Unity.

**Warning:** The tile map editor does not support Undo! (Again, sorry!)

## Selecting map regions to paint

To change the map, select the map in the Hierarchy. You can then drag the mouse to select regions of tiles. The selected region is displayed in green. You can now change what’s in that region using any of the following keyboard commands:

* **Enter**/**return**  
  Paints every tile in the region with the tile type selected in the TileSet (see below)
* **Shift-Delete**  
  Clears the tiles, turning them into completely blank space. These tiles won’t even be drawn.
* **Copy** (control-C or Apple-C)  
  Copies all the selected tiles into the clipboard
* **Paste** (control-V or Apple-V)  
  Paints the region of tiles in the clipboard onto the map. **Important:** the upper-left portion of the painted region will be the upper left corner of the selected region, but width and height of the selected region are ignored. It just paints whatever tiles are in the clipboard.

## Selecting tile types to paint into those map regions

To select a type of type to paint, use the **Go to tile** popup menu from the **TileSet** component. Whatever tile is selected here will be the tile type that gets painted when you hit **Enter** or **Return** in the TileMap itself.

# Defining new tiles

The tile set used here[[2]](#footnote-2) contains many tiles that aren’t currently used. To use a tile in the tile map or for a TileSprite object, you need to first name it so that it appears in the popup menus for selecting tiles. To do that, go to the **Tile Set** component in the **Map** game object, and move to an unnamed tile by holding down control and alt/option, then using the arrow keys to move about the tile set. When you see a tile you want to name, press the **Define attributes** button, and fill in the name, type, and optionally the width and height (if it’s part of a multi-tile object). Note that for multi-tile objects, you should select and name the top-left tile only.

The following is the complete tile set that’s available. Note that there are lots of tiles (e.g. for outdoor scenes) that aren’t currently defined.



1. Usually here means that if you’ve selected a sprite object already, then you can select a different one by clicking on it. Sorry; that was the best I could do given the tools that Unity provides. And sorry this doesn’t work on the Mac; the Mac version doesn’t seem to support the OnSceneGUI() call, or at least never calls it in practice. [↑](#footnote-ref-1)
2. Designed by Silveira Neto. See <http://silveiraneto.net/2009/07/31/my-free-tileset-version-10/> [↑](#footnote-ref-2)