Creating Tilesets

1. Find or create the images you will use for your tiles.

You can separate pre-existing tilesets into individual images using programs like Gimp.
A. Load a tileset into Gimp.
B. Click and drag from the vertical ruler to place guides between each column of tiles.
C. Click and drag from the horizontal ruler to place guides between each row of tiles.
D. Go to Filters -> Web -> Slice…
E. Choose the desired output folder, image type, and name prefix. Do not add spacing.
F. Click OK.

2. Open Texture Packer or your preferred alternative and add your tile images.

3. Under layout, disable Auto Alias and Trim.

4. Change Border Padding and Shape Padding to 0.

5. Change Extrude to 1.

6. Export your tileset.

7. Open Tiled and load a map or create a map; set the tile size to the original size of the tiles before extrusion. A 32x32 tile with an extrusion of 1 is 34x34 pixels, however the tile size in Tiled should be set to 32x32.

8. Go to Map -> New Tileset… and browse to your new tileset.

9. Set Tile Width and Tile Height to the original size of your tiles before extrusion.

10. Set Margin to 1 and Spacing to 2.

11. Set Drawing Offset X and Y to 0.

You are now free to create your map. The Tileset window in Tiled will display the extruded tiles correctly, and Corona will display the tiles with minimal edge artifacts even when scaling is used. Tilesets without any extrusion will work (Margin and Spacing should be set to 0), but they will often show flickering edge boundaries in Corona, particularly if MTE's blockScale is set to anything other than their native size.

The latest daily builds of Corona SDK add nearest-neighbor OpenGL rendering, which may alleviate the need for extruded tilesets if a retro 16-bit graphics style is desired.

Map Structure Overview

Maps for the Million Tile Engine are created in Tiled in more or less the same way as tilemaps meant for other systems. Tiles and objects are arranged in layers, which together make up the tile map. MTE, however, also groups layers into levels.

You can think of a map like a multistory building. Each story of the building is like a level in MTE. Each story contains a floor to walk on, space to move through, objects to interact with, and a ceiling above. Likewise, each level in MTE will usually contain a ground layer, a sprite layer, an object layer (which may also be the sprite layer), and a roof or sky layer. Like the stories of a building these levels are arranged one on top of the other to build a map with depth.

Object layers are created in Tiled. Sprite layers are tile layers or object layers with a property named “spriteLayer” set to “true”. Every level should have one sprite layer and one object layer, or one object layer also serving as a sprite layer. A level can have as many tile layers as necessary.

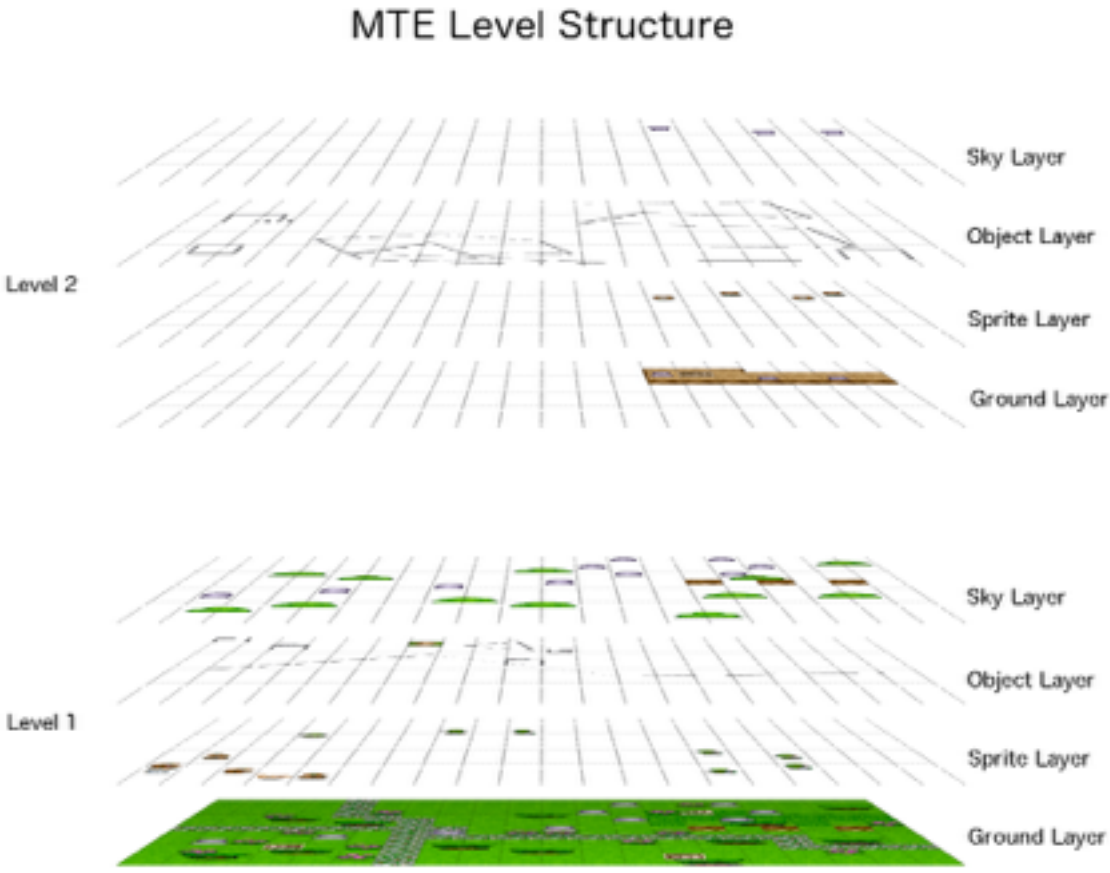
The Million Tile Engine references layers and levels by number. Levels do not exist in Tiled as an actual unit of organization, rather they are created at runtime when the engine reads a layer’s “level” property. Levels must be numbered, they cannot be named. Levels are not necessary if their functionality is not needed. Layers may be given a name in Tiled, however the engine will assign them a number in ascending order at runtime and will reference them by this number.

The MTE functions which take a layer or a level as an argument are expecting a number, not a string or an object reference.

A tile map seen from above may appear something like this.

You can see that it contains grassy ground with sprites standing over it, as well as tall trees and stone monoliths over the top of the sprites. Objects from Tiled are also present in an object layer. In the upper right is a raised area of dirt and rock with more sprites as well as stone pillars and more Tiled objects.

The map is in fact organized into two levels of four layers each.



Each of the first four layers have a property named “level” with a value of 1, and each of the top four layers have a property named “level” with a value of 2. The sprite layer and object layer of a level are easily retrieved programmatically with MTE function calls, simplifying the task of moving sprites from level to level or detecting only the objects on the same level as a sprite.

Getting Started

So you've created or downloaded a tileset, you've put together a map in Tiled, and you've exported that map as a json file. What next? This document assume that your map is named "myMap" and your tilesets are "myTileSet1.png" and "myTileSet2.png" Substitute your own files as necessary.

Create a Project

1. Open Corona Simulator and click the New Project button.
2. Name your app, choose the Blank template for now, and set your screen size and default orientation as you desire.
3. Click next, choose a folder, and click create.
4. Click Show in Finder.

Setup your Project Folder

1. Open the base directory of your project. You will see three files: build.settings, config.lua, and main.lua.
2. Copy mte.lua into the base directory of your project for now.
3. Copy your json map file into the base directory of your project for now.
4. Copy the map's tilesets into the base directory of your project for now.

Loading your Map

1. Open main.lua in your IDE or editor of choice.
2. Require the mte engine file and create an engine instance:
   ```
   local mte = require(mte).createMTE()
   ```
3. Add your tilesets:
   ```
   mte.loadTileSet("myTileSet", "myTileSet.png")
   mte.loadTileSet("myTileSet2", "myTileSet2.png")
   ```
4. Load the map:
   ```
   mte.loadMap("myMap")
   ```

Display your Map

1. Display your map at location (10,10) with a blockScale of 32:
   ```
   mte.setCamera({locX = 10, locY = 10, blockScale = 32})
   ```