PNG Map Editor

What you'll need:

- **A PNG image that represents your room.** Completely white pixels (rgb = #FFFFFF) represent empty space, and you're free to map other colors to your in-game objects and tiles.
- **A "Header" and "Footer" file (textual).** GameMaker: Studio stores rooms in XML format, but this tool only generates the `<instances>…</instances>` and `<tiles>…</tiles>` parts, so it pastes the contents of the Header and Footer files before and after (respectively) its output in the Output file in order to make a complete room file. You can either use the default Header and Footer files you got with the program or make your own (just open one of your rooms in any text editor and copy its contents).
- **A textual configuration file.** Typically, it will be in the following format:

```
SET_HEADER_PATH E:\PNE\header.txt
SET_FOOTER_PATH E:\PNE\footer.txt

SET_RESOLUTION 32
SET_MAX_SCALE  32

0x00, 0x00, 0x00 -> INST(obj_earth_block; CENT, STRC)
```

  The first two lines set the paths to the Header and Footer files. They can be absolute paths, starting with a drive letter, or relative to the working directory of the executable.

  The second two lines set the Resolution and MaxScale attributes. Resolution describes how "large" one pixel of the Input image is. For example, when it's 32, one pixel of the Input image maps to a 32×32 area in-game (in GameMaker units/pixels). MaxScale puts a limit on how much placed objects can be stretched (if they can be stretched at all). For example, if you have a 3×3 area of black pixels in your Input image, and black color is mapped to an object that can be stretched, it will be replaced by a single object with xscale and yscale set to 3, provided that MaxScale is at least 3.

  The final line describes a "pixel mapping". Basically, it tells the program which colour maps to which GameMaker object and how that object behaves. Its format is the following:

```
R, G, B -> INST(gm_name; flag1, flag2, …)
```

  R, G and B must be integers (decimal, octal prefixed with 0 or hexadecimal prefixed with 0x) in range 0-255. "gm_name" is the name of your object in GameMaker. After the name we have a semicolon and then a comma-separated list of optional flags (omit the semicolon if you want no flags).

  The following flags exist:

  - **CENT:** The object uses a centered sprite. Without this flag, origin at (0,0) is assumed.
  - **STRC:** The object can be stretched. Without this flag, a contiguous area of pixels mapping to this object will be replaced with multiple objects (1 per pixel).

Note: This tool only works properly with objects whose sprites have width and height equal to the Resolution attribute set in the Configuration file.

Note: Support for mapping pixels to tiles is planned.

How to use:

1. Prepare all the files described above.
2. Run the application.
3. The application will ask you for the path to the Input image, to the Configuration file and to the Output file. All paths can be either absolute with drive letter prefix or relative to the executable. The output file will be created if it does not already exist (in which case it will be overwritten) but its directory must exist.
4. If the program outputs no exceptions in the console, you'll have an output file ready for use. Otherwise, you'll have to fix the errors (descriptions will be given in the console) and start from step 1.

How to import rooms into GameMaker:

1. Start GameMaker.
2. Create a new room (name it however you like).
3. Close GameMaker.
4. Find your project directory and open your newly created room in any text editor.
5. Replace all contents of the opened file with the contents of the tool's Output file and save the changes.
6. Start GameMaker again. You can now use the generated room.

Contact the author:

For any questions or feedback, contact me through e-mail at jovanbatnozic@hotmail.rs

Or on GMC at https://forum.yoyogames.com/index.php?members/surgeon_.911/

Or on GitHub at https://github.com/Surgeon- (I don't look at GitHub often though).