

## L3tt3rM4pp3r version: 2.0

### About

L4tt3rM4pp3 is a GPU accelerated plugin for Adobe After Effects, allowing artist to display video frames as textmode art. Actually, it's more than that.

Now you basically can use any set of symbols to create your own unique textmode styles, be it CGA BIOS font, Katakana, Klingon or your custom drawn symbols - just put them into proper texture and plugin will do the rest.

You can use the symbols that come with the plugin, draw the texture by yourself, or L3tt3rM4pp3r comes with two scripts for Adobe Photoshop, that are simplifying the process of creating maps that plugin can use.

- **layers2ramp.jsx** analyzes and combines set of layers in PSD file into texture that will be compatible with the plugin
- **bmf2ramp.jsx** lets you to make use of any Truetype or Opentype font via [Bitmap font generator tool](#) - once you exported original font file into BMF, you can parse it and make a texture out of it
- **atlas2layers.jsx** slices current document to layers, which can be used with layers2ramp script later

Example set of mapping textures is also included.

# Installation

L3tt3rM4pp3r 2 can be installed by copying the contents of the archive into one of the After Effects plug-in folders. By default, the plug-ins folder is in the following location, according to [Adobe After Effects online help](#):

## **Windows:**

**\Program Files\Adobe\Adobe AfterEffects[version]\Support Files\Plug-ins**

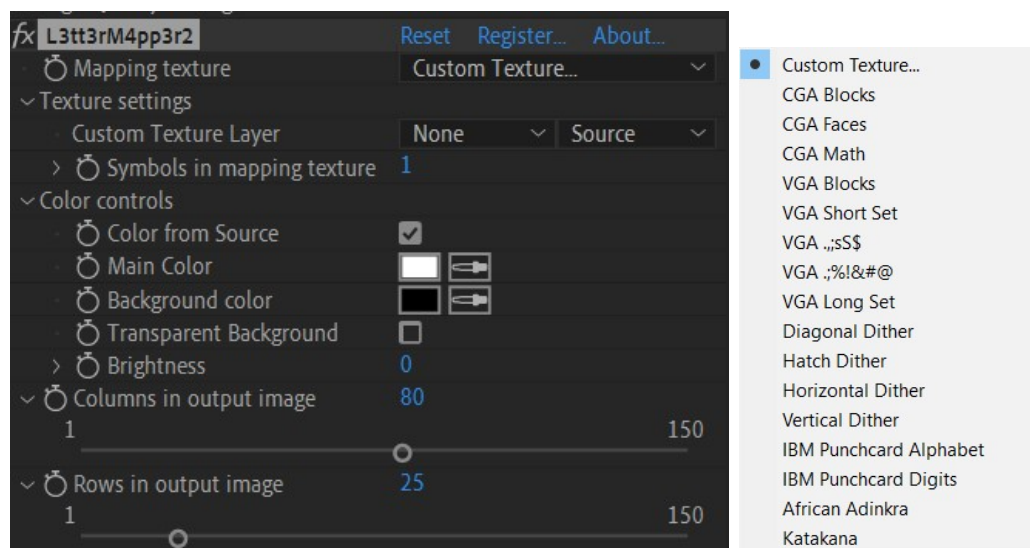
## **Mac:**

**/Applications/Adobe After Effects[version]/Plug-ins**

Scripts can be installed from the Scripts folder of the archive wherever you want, and run Photoshop .jsx scripts from it when you feel like making some new texture maps with symbols using either Photoshop PSD file with layers or BMF file.

Evaluation version of L3tt3rM4pp3r 2 doesn't have any time limitations, and outputs random pink squares on screen. You can register/buy the plugin by pressing the **Register...** link, which is visible in the demo version. Please note that for After Effects CS6 on Macs it is required to restart the host after entering your registration number, to prevent rendering glitches.

## Plugin's parameters and algorithm



Let's see what parameter L3tt3rM4pp3r 2 has and how it works:

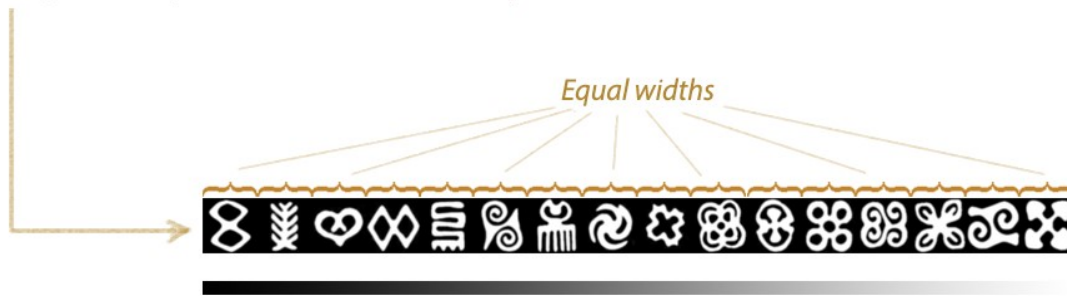
To use L3tt3rM4pp3, apply it to layer; also, you need to either keep one of predefined textures (like CGA Blocks), or add an image of **Mapping texture** to the same composition and hide its layer. It doesn't matter if the **Custom Texture** layer is visible or not.

What the plugin basically does is replaces the content of the layer it was put on with symbols from the **Mapping texture**, using the source layer as a mask.

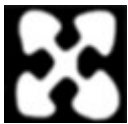
Here's an example of an actual mapping texture.

In fact, it is a set of white glyphs (symbols) on black background, placed from left to right, depending on the area size of a particular symbol (the more to the right, the “fatter” symbol is. You can think of it as the linear gradient, actually.

This particular map is called “Adinkra\_16”. It was made out of public domain [Adinkra](#) symbols using **layers2ramp.jsx** script. There are 16 symbols in it, and we'd recommend putting the amount of symbols into the texture name for overall convenience. That's what our Photoshop scripts do as well.



One single symbol.  
This is on the left in the row,  
smallest amount of white pixels per symbol area



Another single symbol.  
This one is at the right of the whole row,  
maximum count of white pixels per symbol area.

Plugin creates a picture of **Columns in output image \* Rows in output image** blocks, same way as "Pixelize" effect works.

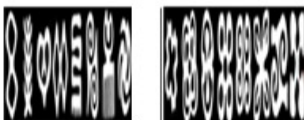
Then, it measures luminosity of each block and draws some area of the **Mapping texture** on top of it. The brighter luminosity is, the more to the right symbol from the row will be picked.

As you can load different images into mapping texture, plugin needs to know what exactly area of the texture to use to pick up the symbol correctly.

So, when you're setting **Symbols in mapping texture** to 16, plugin will divide the whole picture by 16 columns of equal width and will treat each column as symbol.

If your source map is made of 16 symbols, all should work as intended. If there's more/less, wrong areas will be used.

E.g., if you'll set this param to 2 for the same map, entire picture will be mapped using these two symbols (two halves of the entire map actually):



So, better be precise with your **Symbols in mapping texture**.

The rest of params are there to tune colors:

**Color from source** - use color from the source layer

**Main Color** - color of mapping symbols

**Background Color** - background behind the mapping symbols

**Transparent Background** - mapping symbols are over transparent background

**Brightness** - tweak input brightness

## Included Photoshop scripts

Please note that due to the nature of Photoshop scripting, it is not possible to do several things fast, so these two scripts may take a while to work (around 5 minutes for big sets of symbols on a modern workstation)

### layers2ramp.jsx

The script analyzes and combines set of layers in PSD file into texture that will be compatible with the plugin.

Create PSD file with one white symbol on black background per layer. Then save the PSD file and run the script. The script will ask you to select the source PSD file, load it, analyze glyph data and create new texture map - PSD file with symbols placed from left to right, sorted by overall symbol area.

### bmf2ramp.jsx

The script helps to export any True Type or Open Type font to L3tt3rM4pp3r texture map via BMF raster font file. To do so,

- open any TTF/OTF font to BMF using [Bitmap font generator tool](#)
- select the set of symbols you'd like to export, or just select all
- be sure you've set font descriptor to XML and texture format to PNG in tool's export options
- run the script and open the BMF file
- the script will analyze glyph data and create new texture map - PSD file with symbols placed from left to right, sorted by overall symbol area
- you can then edit the file and merge all layers/save when ready

### atlas2layers.jsx

Slices current document to layers, which can be used with **layers2ramp** script later.

Script is applied to active document, so with your atlas opened in Photoshop, run this script, enter atlas rows/columns amount and be patient for a little while.