GAME BOY SW TUTORIAL

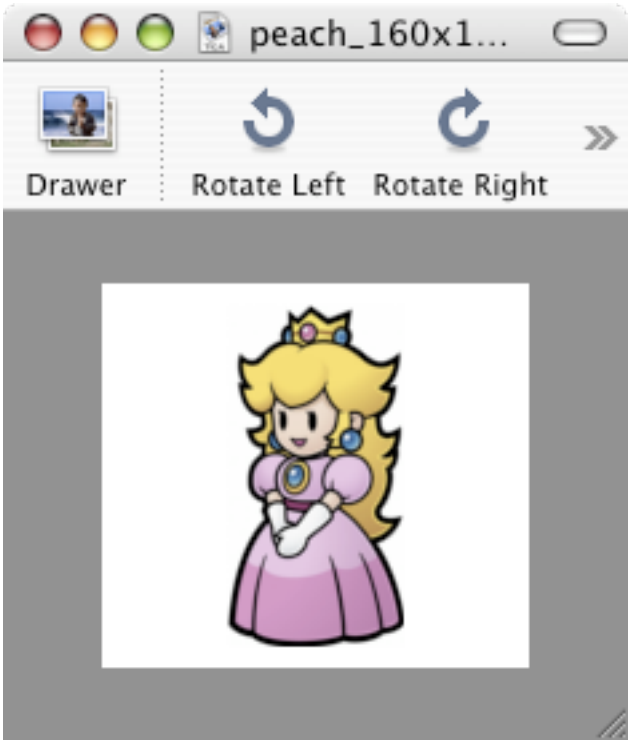
This document will walk you through the basics of getting your first demo running on an original Nintendo Game Boy (DMG, Pocket, Color & compatibles).

It is recommended to become familiar with the architecture of the machine itself before starting to write any line of code. Details on the processor, peripherals and I/Os can be easily found today on a wide variety of websites and also in the famous PAN DOCS ("Everything You Always Wanted To Know About GAMEBOY").

prerequisites:
- TGA2BG utility binary
- WLA-DX gb-z80 assembler binaries [wla-gb_9.5a_xxx_yyy.zip]
- gb-z80 assembly code templates [gbz80_asm_020710.xxx]

step 1

Create a 160x144 TGA picture which can be processed by the TGA2BG utility (please refer to the TGA2BG manual for detailed specifications).



step 2

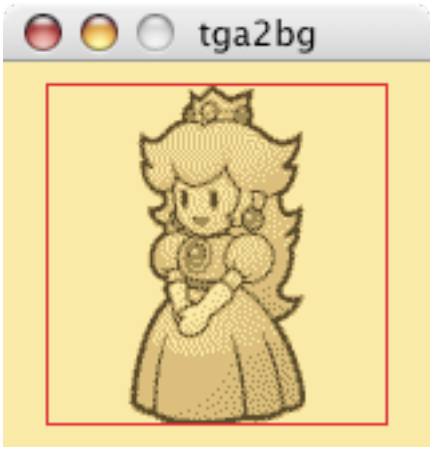
The TGA2BG utility will convert the TGA picture to a 128x128 tiles map.

```
tga2bg peach_160x144.tga peach_128x128.inc
```

then press 'w' and the following message will be displayed:

```
tile data written to peach_128x128.inc
```

Remember, because of the Game Boy 256 tiles limitation, only the picture area which fits into the red square (128x128 pixels) will be displayed on the console screen. (press the space bar to toggle the red area on and off).



step 3

Preparing the build environment. The gbz80_asm_020710.xxx contains a build script for the win32 platform (*.bat) and one for Mac OS X (*.sh) as well as assembly code templates for a basic demo. Users of other UNIX flavors can re-use the Mac OS X build script and compile WLA-DX to run on their own hardware/OS combination.

- Uncompress the archive, you will obtain a gbz80_asm_020710 folder.
- Depending on which OS you are using, add the wla-gb_9.5a_ppc_osx.zip or the wla-gb_9.5a_x86_win32.zip archive content to the gbz80_asm_020710 folder.
- Replace the file mario_128x128.inc file by the newly created peach_128x128.inc. The gbz80_asm_020710 folder should now look like this:

```
gb_as_ppc_osx.sh
gb_as_x86_win32.bat
hw.inc
logo.inc
peach_128x128.inc
rom.lds
rom.s
rst.inc
wla-gb_9.5a_ppc_osx      [MacOSX only]
wla-gb_9.5a_x86_win32    [Win32 only]
```

- edit rom.s and change the content of line 159.

[original]
.include "mario_128x128.inc"

[new]
.include "peach_128x128.inc"

note: the mario_128x128.inc file and the original rom.s file works perfectly fine and compiling them will create a ROM with a mario on the screen. The process here is showing how to change the content of the screen on top of the basic compilation process.

step 4

Building the ROM. From a terminal or a command prompt use the following commands:

[MacOSX / UNIX]
./gb_as_ppc_osx.sh

[Win32]
gb_as_x86_win32.bat

a rom.gb file will be created. This is the actual ROM file.

step 5

Proving out the result. Using your favorite Game Boy emulator or any type of programmable cartridge, load and run the rom.gb file.