

Link : <https://oblerion.itch.io/gba-engine> ver a1.8 2024

# Engine GBA

The gameboy advance like engine  
by Magnus Oblerion



All rights reserved 2024

Link : <https://oblerion.itch.io/gba-engine> ver a1.8 2024

## Describe :

2d game engine made with C, raylib.h and lua.h .

egba = Tic80 api + 32bits upgrade + modern GPU optimization.

It's the synthesis of 7 years of game and framework creation,  
which is why (exceptionally) the sources are closed.

## Features:



- UI Project browser (add/delete/load project)
- UI Project (edit/debug/run/export/import)
- UI Sprite viewer (view/save/load-delete palette/load spritesheet)
- UI Script viewer (view)
- locked cartbridge
- Color Theme (white/black/blue/red/green/yellow)
- Run egba and lua in CLI
- Save-state 100 bools, 50 string, 50 number
- Your game can run other local game
- Read other save-state

## Spec:

- 100 projets max in browser
- 5 palette color : 32 colors + alpha
- 256 sprite : 16x16
- lua script : 64000 char

## Fast start :

### Setup new project

- Download palette image .png <= 32 colors [lospec.com](https://lospec.com)
- on browser touch ,  it create new.lua and new.png
- click on « new » project, click on  and sprite button
- drag and drop palette image to window
- save, quit and rename new .lua/.png with your projet name

Link : <https://oblerion.itch.io/gba-engine> ver a1.8 2024

## Debug (GUI)

- load projet with browser
- press space for start/stop debug

## Create EGBA from script/sprite

- After load projet , « -> » button or ctrl + e
- ctrl + l for locked egba (can't import lua/spr from egba)

## Import sprite/script from EGBA

After load projet , « <- » button or ctrl + i

## Run EGBA (GUI)

After load projet, press enter for start/stop running EGBA

## Debug (CLI)

- open cmd, tape it

```
egba.exe project.lua
```

it use .png for load color and sprite

## Run (CLI)

- open cmd, tape it

```
egba.exe project.egba
```

## Load spritesheet

- drag and drop 256x256 img, it to browser → project → spr
- drag and drop 256x257 img, it to browser → project → spr

Link : <https://oblerion.itch.io/gba-engine> ver a1.8 2024

## Lua api :

### Main

#### loop function

```
function EGBA()  
-- your code run 60/seconds  
end
```

#### log

```
trace("hi")  
-- print hi in console (only string)  
  
local v=2 print("value :",v)  
-- lua print function can be used to view variable value in console
```

#### palette swap

```
pal(2)  
-- id 0 -> 4, change curant palette  
-- for all function after with id color
```

### Graphics

#### clear screen

```
cls(0)  
-- clear screen with id color 0
```

#### print text

```
text("hello world",23,23,0,16)  
-- x 23, y 23, id color 0,font size 16
```

#### draw rectangle

```
rect(23,23,50,50,2)  
-- draw fill rectangle in x 23, y 23, width 50, height 50, id color 2  
rectb(23,23,50,50,2)  
-- draw line rectangle in x 23, y 23, width 50, height 50, id color 2
```

#### draw pixel

```
pix(10,10,2)  
-- draw pixel in x 10,y 10, id color 2
```

#### draw sprite

```
spr(1,23,23,2)  
-- draw sprite id 1,x 23,y 23,scale 2
```

## Input

### keyboard

```
if btn(0) then
-- if key down is down
elseif btn(1) then
-- if key up is down
end
if btnp(2) then
-- if key left is pressed
elseif btnp(3) then
-- if key right is pressed
end
-- id 0 : up, 1 : down, 2 : left, 3 : right, 4 : x, 5 : c
-- btn -> btn is down
-- btnp -> btn is pressed
```

### mouse

```
local x,y,btnl,btnm,btnr = mouse()
-- x = x mouse
-- y = y mouse
-- btnl = true/false mouse left button
-- btnm = true/false mouse mid button
-- btnr = true/false mouse right button
```

## Save-state (.sav file)

This is static data, it keep value when quit.

### Write save

```
wsave_num(id,100)
-- id 0 → 50
wsave_string(id,"name")
-- id 0 → 50, string 35 char
wsave_bool(id,true)
-- id 0 → 100
```

### Read save

```
local i_num = rsave_num(id)
-- id 0 → 50
local i_string = rsave_string(id)
-- id 0 → 50
local i_bool = rsave_bool(id)
-- id 0 → 100
```

Link : <https://oblerion.itch.io/gba-engine> ver a1.8 2024

## Interverse

Run an other game, read extern save.

### Run game

```
run("projet.egba") -- work with .lua
```

### Read extern save

```
-- if target is projet.sav
local numb = rsave_numb_ext("projet",id)
local nstring = rsave_string_ext("projet",id)
local nbool = rsave_bool_ext("projet",id)
```

### Entry point

Create projet.entry for run (projet.lua / projet.egba) at start.

## Music

Load / play .wav .mp3 .ogg

### Load directory

```
music_loaddir("asset") -- load all .wav .mp3 .ogg in dir
```

### Play sound

```
music_playsound("lazer") -- play lazer.wav
```

### Play music

```
music_playmusic("music_intro") -- play/resume music_intro.mp3/.ogg
```

### Pause music

```
music_pausemusic("music_intro") -- pause music_intro.mp3/.ogg
```

### Stop music

```
music_stopmusic("music_intro") -- stop music_intro.mp3/.ogg
```

## Table index

Describe :.....	2
Features:.....	2
Spec:.....	2
Fast start :.....	2
Setup new project.....	2
Debug (GUI).....	3
Create EGBA from script/sprite.....	3
Import sprite/script from EGBA.....	3
Run EGBA (GUI).....	3
Debug (CLI).....	3
Run (CLI).....	3
Load spritesheet.....	3
Lua api :.....	4
Main.....	4
loop function.....	4
log.....	4
palette swap.....	4
Graphics.....	4
clear screen.....	4
print text.....	4
draw rectangle.....	4
draw pixel.....	4
draw sprite.....	4
Input.....	5
keyboard.....	5
mouse.....	5
Save-state (.sav file).....	5
Write save.....	5
Read save.....	5
Interverse.....	6
Run game.....	6
Read extern save.....	6
Entry point.....	6
Music.....	6
Load directory.....	6
Play sound.....	6
Play music.....	6
Pause music.....	6
Stop music.....	6