HAM Tutorial :: Day 5 :: Sprites

I think it's time to start learning about sprites, don't you? 'What is a sprite?' you ask. Well, I would think you already know, but just in case you don't, a sprite is just an image on the screen (except the background, although a sprite could be that big). Any character, coin, etc. on the screen is a sprite.

Just as you saw with backgrounds in Day 4, a sprite must be in the proper format. Again, a sprite can be from 8x8 up to 64x64 pixels. There can be up to 128 sprites on the screen at a time. They can share one 256 color palette or there can be up to 16 different 16 color palettes that a sprite can use.

As I mentioned in Day 4, I will work exclusively with bitmaps. For this tutorial I have created a sprite which is just a blue ball called blue_ball.bmp. Copy the file to your **gfx** subdirectory and type:
gfx2gba -D -fsrc -pblue_ball.pal -t8 blue_ball.bmp

This will create two files: **blue_ball.pal.c**, **blue_ball.raw.c**

Again, two arrays will be created. The palette will be **blue_ball_Palette** and the image will be **blue_ball_Bitmap**.

For today's code we'll just display the sprite on the screen. Later we'll work on moving it around the screen. (I hope that you all know what global variables are.)

NOTE: You'll notice that when the ball is displayed on the screen you see the white background that is part of the original image. Later on when we cover alpha blending, you'll learn how to show only the blue ball without the white background.

```
// The Main HAM Library
#include <mygba.h>

// Graphics Includes
#include "gfx/blue_ball.raw.c"
#include "gfx/blue_ball.pal.c"

// Global Variables
u8 theball; // Sprite object number of the ball

// Function: main()
int main()
{
    // Initialize HAMlib
    ham_Init();

    // Setup the background mode
    ham_SetBgMode(4);

    // Initialize the sprite palette
```

Poll

Discussions

Graphics FAQ

GFX2GBA Readme

Translations

Downloads

Games

Projects

Credits

Links

Support

```
    ham_LoadObjPal((void*)blue_ball_Palette, 256);

    // Create the sprite and store the object number
    theball = ham_CreateObj((void*)blue_ball_Bitmap,0,2,
                            OBJ_MODE_NORMAL,1,0,0,0,0,0,0,110,50);

    // Draw the sprite
    ham_CopyObjToOAM();

    // Infinite loop to keep the program running
    while(1) {}

    return 0;
} // End of main()
```

Code Explanation

u8 theball
A u8 is defined in mygba.h as an unsigned char. Each object that is created in memory has an object number (from 0 to 127). For this example, there is just one sprite so I simply store the object number in *theball*. Later on I will use arrays for sprites because if the sprite rotates or turns at all, it is easy to load the different frames into the array and use this for animation. This may seem somewhat confusing so I will explain it more when we do our first project.

ham_SetBgMode(4)
For this example we'll use Mode 4 although it doesn't really matter. Sprites are separate of BGMode *except* that there is less sprite storage space in certain BG Modes. However, HAM handles the trouble there, no matter what BGMode you are in. You'll just have less available OBJ RAM in some modes.

theball = ham_CreateObj(...)
ham_CreateObj is used to create a sprite in the HAM system and return the reference index for further operations on the sprite. It is not very difficult to understand but there is a lot of information that you must pass to the function. The first thing is the array associated with the image that you want to display. Next is the size and shape of the object. Check the chart to see what your choices are. In this case, I am using an 32x32 pixel image, so I pass it 0 and then 2. What this means is that there are 4x4 8 pixel blocks. Think about that for a minute to make sure you understand what I mean. (**NOTE**: Instead of passing 0,2 you could use OBJ_SIZE_32X32.) Next is the object mode. For now, I am just using OBJ_MODE_NORMAL. Next is the color mode. For this sprite I have one 256 color palette, so I pass it 1. Next is the palette number. Because I am using color mode 1, you just pass it 0. If you were using 16 separate 16 color palettes, you would pass it the palette number which corresponds to your image (0-15). Next are mosaic, horizontal flipping and vertical flipping. I am not using any of them, so they are all 0. Next is the priority of the object, in this case it's 0. Finally you pass it the X and Y position on the screen.

ham_CopyObjToOAM()
This function is used to load all of the objects to OAM. This basically means it

draws the objects.

Well, another small step today but a major concept to understand. Try creating your own sprite, convert the file with **gfx2gba** and see if you can display it. If you want to be really creative, make the sprite a size other than 32x32 pixels and see if you can get it to show up properly.

Download Code

NOTE: You may need to Right-click and choose Save As.

HAM Version 2.40 And Higher
Download All Files In One Zip: Day5_Sprites.zip

View Demo Now

Discuss Day 5

<< HOME >>