



Desarrollo para Nintendo DS

Make Classic Games



Índice

1. Introducción
2. Nintendo DS
 - a. Modelos
 - b. Especificaciones Técnicas
 - c. Flash Carts
3. Arquitectura
4. Juegos para Nintendo DS
5. Desarrollo Homebrew para Nintendo DS
 - a. DevkitPro
 - b. DESmume (emulador)/no\$gba
6. Desarrollo de videojuegos para Nintendo DS
 - a. Hola mundo
 - b. Texto por pantalla
 - c. Fondos
 - d. Sprites

Introducción

Esta presentación, muestra como desarrollar videojuegos para el sistema portátil Nintendo DS.



En este caso, se usarán herramientas independientes para poder desarrollar el juego para este sistema. NOTA: no está licenciado por Nintendo.

Nintendo DS

Nintendo DS (Dual Screen), es una consola desarrollada por Nintendo y salió al mercado el año 2004, que es la sucesora de la Game Boy Advance. Pertenece a la séptima generación de consolas.



Una de las principales características de esta consola fue la doble pantalla; siendo la pantalla inferior táctil. Tuvo varias revisiones y su sucesora, la Nintendo 3DS era compatible con esta.

Nintendo DS

Nintendo DS tuvo varias revisiones de la misma; algunas ampliaban las capacidades de la consola:

DS (2004)



DS Lite (2006)



DSi (2008)



DSi XL (2010)



Nintendo DS

La sucesora de la Nintendo DS, fue la Nintendo 3DS en 2011; que tuvo una gran mejora como el uso de 3D sin necesidad de utilizar gafas. Además, era compatible con los juegos de Nintendo DS.



Nintendo DS

Las especificaciones técnicas de la Nintendo DS son:

CPU	ARM9 (ARM946E-s) a 67Mhz	Utilizado para software NDS
	ARM7 (ARM7TDMI) a 33Mhz (compatible con GBA)	Utilizado cuando se juega como GBA
Memoria	4MB RAM	16MB para modelos DSi y DSi XL
	656KB VRAM	Usado para vídeo
GPU	Integrado en el propio Chip	Tiene capacidad tanto para 3D como para 2D
Pantalla	2 pantallas de 256x192 píxeles	Los modelos DSi y DSi XL ampliaban tamaño de pantalla.
Sonido	Sonido Estéreo y micrófono	
Red	Tarjeta de red Inalámbrica	
Slots	2 Slots para tarjeta (DS Arriba y GBA abajo)	En modelos DSi y DSi XL la ranura GBA, se cambiaba por una tarjeta SD.
Cámaras	2 cámaras de 0.3 Megapíxeles	Solo en modelos DSi y DSi XL.

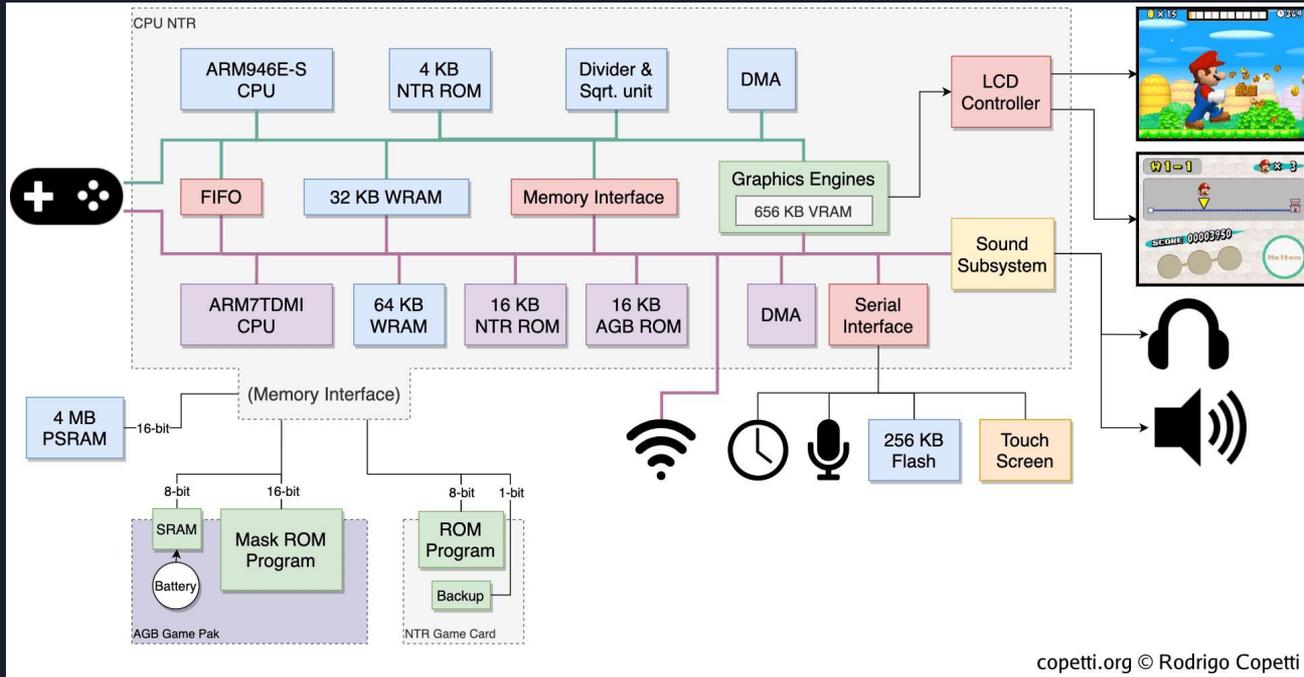
Nintendo DS

Para poder ejecutar los juegos de Nintendo DS, se utilizan unas tarjetas de memoria con formato “Nintendo DS Game Card” que tienen una capacidad desde 8MB hasta 512MB.

La ranura inferior, en los modelos DS y DS Lite, permiten introducir cartuchos para Game Boy Advance.



Arquitectura



Fuente: Rodrigo Copetti

Juegos para Nintendo DS





Desarrollo Homebrew para Nintendo DS

- No se puede desarrollar oficialmente para Nintendo DS, sin una licencia de Nintendo.
- Sin embargo, usando herramientas de terceros, podemos crear nuestros propios juegos de este sistema; para uso personal o no comercial.
- Utilizando estas herramientas y los llamados Flash carts, podemos crear nuestros propios juegos para Nintendo DS.
- El conjunto de herramientas que vamos a utilizar en este caso, es *DevKitPro*.

Desarrollo Homebrew para Nintendo DS

- Las Flash Carts, son tarjetas de memoria con el formato de nintendo DS, que incorporan un lector de tarjetas micro SD y nos permiten cargar juegos personalizados. Sin embargo, no es necesario para desarrollar ya que podemos utilizar emuladores.
- Los más conocidos son los R4 o los M3.



Desarrollo Homebrew para Nintendo DS

DevkitPro, es un conjunto de herramientas y librerías para poder desarrollar en distintos sistemas entre ellos los de Nintendo. Permite desarrollar para:

- GBA
- NDS
- 3DS
- Wii
- WiiU
- Switch



Desarrollo Homebrew para Nintendo DS

Podemos encontrar toda la información acerca de devkit pro en su página web:

- <https://devkitpro.org/>

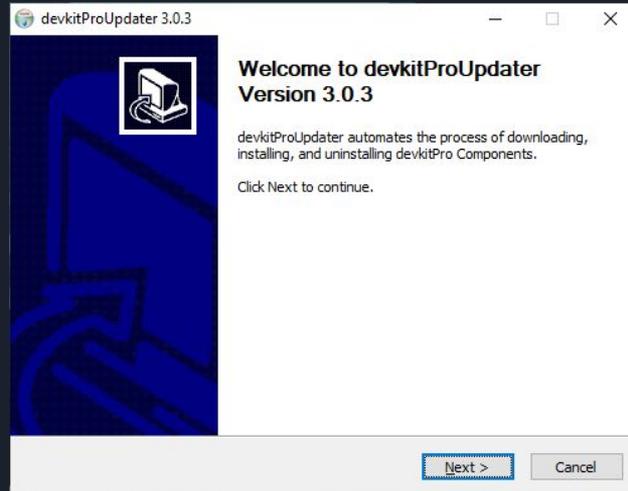


Desarrollo Homebrew para Nintendo DS

Windows

Para instalar en windows, utilizaremos el auto instalador que descargamos desde la página su github:

- <https://github.com/devkitPro/installer/releases>





Desarrollo Homebrew para Nintendo DS

Linux

- Para poder instalar la versión de Linux, necesitaremos instalar una serie de requisitos:

```
apt install wget curl libarchive13 xz-utils make gpg pkg-config
```
- Descargamos el gestor de paquetes “pacman” para devkitpro (si ya se dispone de pacman se puede utilizar este gestor).

```
wget https://github.com/devkitPro/pacman/releases/download/v1.0.2/devkitpro-pacman.amd64.deb
```

- Instalamos el paquete deb (para sistemas basados en debian).

```
dpkg -i devkitpro-pacman.amd64.deb
```

- Ejecutamos la siguiente instrucción:

```
ln -s /proc/self/mounts /etc/mtab
```

- Ejecutamos el gestor de paquetes pacman y seguimos instrucciones para nds-dev

```
dkp-pacman nds-dev
```



Desarrollo Homebrew para Nintendo DS

Linux

- Crear las siguientes variables de entorno:

```
export DEVKITARM=/opt/devkitpro/devkitARM
export DEVKITPRO=/opt/devkitpro
export DEVKITPPC=/opt/devkitpro/devkitPPC
export PATH="${DEVKITPRO}/tools/bin:${PATH}"
```



Desarrollo Homebrew para Nintendo DS

MacOs

- Descargar el instalador de la siguiente dirección:

<https://github.com/devkitPro/pacman/releases/download/v1.0.2/devkitpro-pacman-installer.pkg>

- El resto de pasos, se realiza igual que en Linux.



Desarrollo Homebrew para Nintendo DS

Docker

Podemos utilizar un contenedor, para poder realizar el desarrollo para Nintendo DS; para ello, se ha generado un fichero *DockerFile*, para poder generar una imagen y utilizar un contenedor.

- Clonar el siguiente repositorio:

<https://github.com/zerasul/dockerretro>

- en la carpeta nds, crear la imagen docker:

```
docker build -t nds .
```

- Para compilar cada ejemplo, puede hacerse con el siguiente comando (en windows, cambiar \$PWD por %CD%):

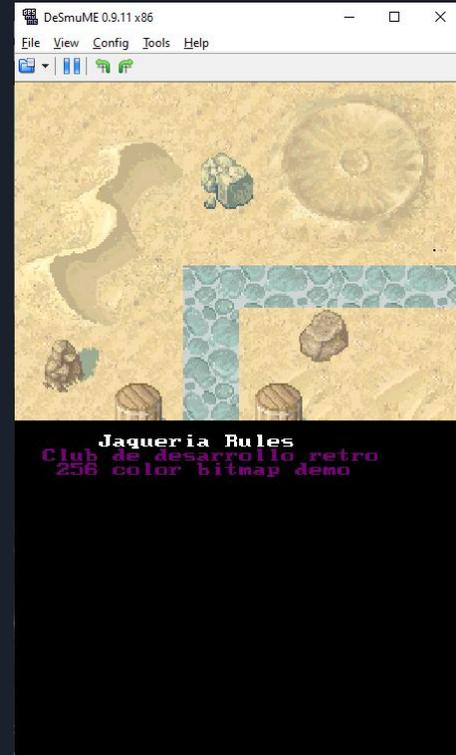
```
docker run --rm -v $PWD:/src/nds nds
```

Desarrollo Homebrew para Nintendo DS

DesMume

DesMume, es un emulador para Nintendo DS, que viene con algunas herramientas para desarrollo; existen versiones tanto en Windows, Linux y Mac; y será el que utilizaremos para probar nuestros juegos mientras desarrollamos.

También puede utilizarse no\$gba como emulador.



Desarrollo Homebrew para Nintendo DS

DesMume

Para instalar este emulador, en cada Sistema;
descargar desde la siguiente dirección:

<http://desmume.org/download/>

windows:

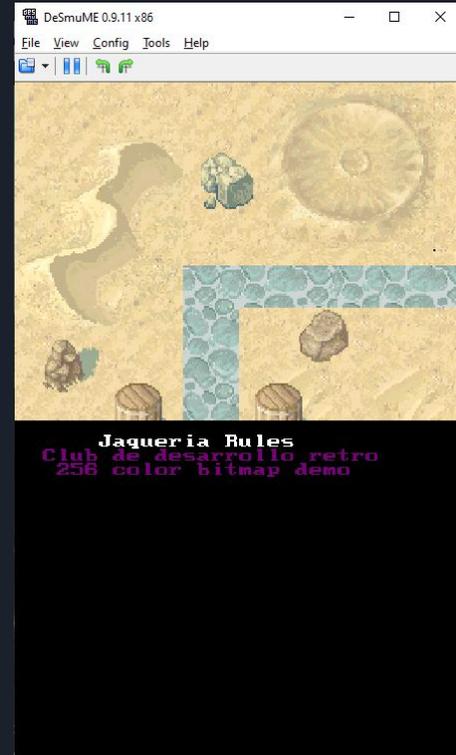
- descargar zip y descomprimir.

linux:

- instalar con apt
- instalar con FlatPak

Macos:

- Descargar y ejecutar instalador



Desarrollo Homebrew para Nintendo DS

no\$gba

Para instalar este emulador, en cada Sistema;
descargar desde la siguiente dirección:

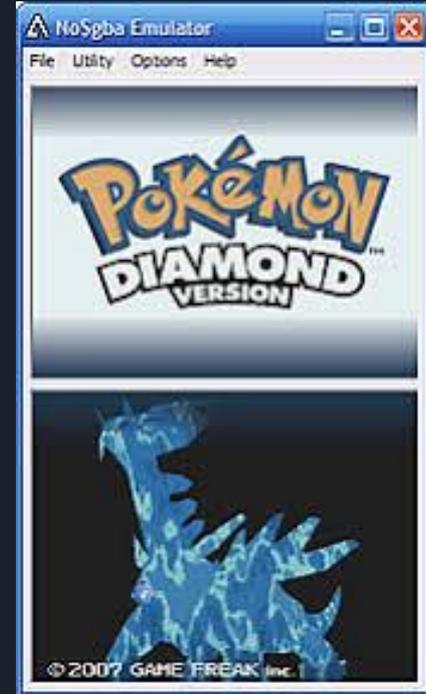
[https://www.nogba.com/no\\$gba-download.htm](https://www.nogba.com/no$gba-download.htm)

windows:

- descargar zip y descomprimir.

linux:

- instalar con apt
- instalar con FlatPak





Desarrollo de videojuegos para Nintendo DS

Ahora que hemos visto tanto la propia Nintendo DS, como las herramientas que utilizaremos para desarrollar nuestros juegos, vamos a ver distintos ejemplos, donde veremos los conceptos necesarios para crear nuestro juego para Nintendo DS.

Vamos a utilizar un repositorio con los ejemplos que veremos en este taller; podéis descargarlo de la siguiente dirección:

<https://github.com/LaJaqueria/nds-examples>

Para compilar cada ejemplo, se puede hacer de la siguiente forma:

Linux y Macos

```
make
```

Windows

```
make #debe estar la variable Path correctamente configurada
```

Docker

```
docker run --rm -v $PWD:/src/nds nds
```



Desarrollo de videojuegos para Nintendo DS

Hola Mundo

```
#include <nds.h>

#include <stdio.h>

volatile int frame = 0;

//-----
void Vblank()
{
    //-----
    frame++;
}
```



Desarrollo de videojuegos para Nintendo DS

Hola Mundo

```
..
int main(void)
{
    //-----
    touchPosition touchXY;

    irqSet(IRQ_VBLANK, Vblank);

    consoleDemoInit();

    iprintf("    Jaqueria NDS\n");
    iprintf("    \x1b[32mwww.lajaqueria.org\n");
    iprintf("    \x1b[32;1mNintendo DS Homebrew\x1b[39m");
}
```



Desarrollo de videojuegos para Nintendo DS

Hola Mundo

```
while (1)
{

    swiWaitForVBlank();
    scanKeys();
    int keys = keysDown();
    if (keys & KEY_START)
        break;

    touchRead(&touchXY);

    // print at using ansi escape sequence \x1b[line;columnH
    iprintf("\x1b[10;0HFrame = %d", frame);
    iprintf("\x1b[16;0HTouch x = %04X, %04X\n", touchXY.rawx, touchXY.px);
    iprintf("Touch y = %04X, %04X\n", touchXY.rawy, touchXY.py);

}

return 0;
```



Desarrollo de videojuegos para Nintendo DS

Hola Mundo

```
irqSet(IRQ_VBLANK, Vblank);
```

Esta función define el controlador de la interrupción lanzada cada vez que se pinta la pantalla (60 veces por segundo).

```
consoleDemoInit();
```

Esta función, inicializa la consola en modo “demo”; activa la pantalla de abajo con el primer fondo en modo texto.

```
iprintf("    Jaqueria NDS\n");
```

iprintf, permite escribir texto por pantalla (debe de activarse el modo texto); además permite escapar caracteres para definir colores, posición, etc...



Desarrollo de videojuegos para Nintendo DS

Hola Mundo

```
swiWaitForVBlank();
```

Bloquea la ejecución, hasta que termine de pintar la pantalla.

```
scanKeys();
```

Escanea las teclas pulsadas.

```
int keys = keysDown();
```

Devuelve una máscara con las teclas pulsadas.

```
touchRead(&touchXY);
```

Devuelve la posición del lápiz táctil (si hay varias posiciones, devuelve el baricentro).

Desarrollo de videojuegos para Nintendo DS

Hola Mundo

Si compilamos y ejecutamos el ejemplo, veremos en la parte inferior el nº d frame y si clickeamos , veremos el centro de dónde estamos pulsando con el lápiz táctil.





Desarrollo de videojuegos para Nintendo DS

Texto

En el anterior ejemplo, hemos visto cómo escribir texto por pantalla. Sin embargo, ha sido solo en 1 pantalla. En este ejemplo, veremos cómo inicializar la consola manualmente para usar las dos pantallas. Veamos un fragmento:

```
PrintConsole topScreen;
PrintConsole bottomScreen;

//Init main and sub consoles to text mode
videoSetMode(MODE_0_2D);
videoSetModeSub(MODE_0_2D);
consoleDemoInit();
vramSetBankA(VRAM_A_MAIN_BG);
vramSetBankC(VRAM_C_SUB_BG);

consoleInit(&topScreen, 3, BgType_Text4bpp, BgSize_T_256x256, 31, 0, true, true);
consoleInit(&bottomScreen, 3, BgType_Text4bpp, BgSize_T_256x256, 31, 0, false, true);

consoleSelect(&topScreen);
iprintf("\n\n\t\tLa Jaqueria\n");
iprintf("\t\x1b[31mTest texto Superior\n");
iprintf("\t\x1b[35mwww.lajaqueria.org");
```

Desarrollo de videojuegos para Nintendo DS

Texto

Veamos las funciones más importantes:

```
videoSetMode(MODE_0_2D);
```

Configura en modo de vídeo de la pantalla superior; dependiendo de esta configuración, se pueden utilizar distintos fondos en distinta forma:

Main 2D engine

Mode	BG0	BG1	BG2	BG3
0	T	T	T	T
1	T	T	T	R
2	T	T	R	R
3	T	T	T	E
4	T	T	R	E
5	T	T	E	E
6		L		

T = Text

R = Rotation

E = Extended Rotation

L = Large Bitmap background

Desarrollo de videojuegos para Nintendo DS

Texto

```
videoSetModeSub(MODE_0_2D);
```

Configura en modo de vídeo de la pantalla inferior; dependiendo de esta configuración, se pueden utilizar distintos fondos de distinto tipo:

Sub 2D engine

Mode	BG0	BG1	BG2	BG3
0	T	T	T	T
1	T	T	T	R
2	T	T	R	R
3	T	T	T	E
4	T	T	R	E
5	T	T	E	E

Desarrollo de videojuegos para Nintendo DS

Texto

```
vramSetBankA(VRAM_A_MAIN_BG);
```

Nintendo DS, tiene la memoria RAM configurada en 9 bancos de memoria. Dependiendo de lo que necesitemos, podemos configurar cada banco para un uso (fondos, sprites,etc...).

Enumerator	
VRAM_A_LCD	maps vram a to lcd.
VRAM_A_MAIN_BG	maps vram a to main engine background slot 0.
VRAM_A_MAIN_BG_0x06000000	maps vram a to main engine background slot 0.
VRAM_A_MAIN_BG_0x06020000	maps vram a to main engine background slot 1.
VRAM_A_MAIN_BG_0x06040000	maps vram a to main engine background slot 2.
VRAM_A_MAIN_BG_0x06060000	maps vram a to main engine background slot 3.
VRAM_A_MAIN_SPRITE	maps vram a to main engine sprites slot 0.
VRAM_A_MAIN_SPRITE_0x06400000	maps vram a to main engine sprites slot 0.
VRAM_A_MAIN_SPRITE_0x06420000	maps vram a to main engine sprites slot 1.
VRAM_A_TEXTURE	maps vram a to 3d texture slot 0.
VRAM_A_TEXTURE_SLOT0	maps vram a to 3d texture slot 0.
VRAM_A_TEXTURE_SLOT1	maps vram a to 3d texture slot 1.
VRAM_A_TEXTURE_SLOT2	maps vram a to 3d texture slot 2.
VRAM_A_TEXTURE_SLOT3	maps vram a to 3d texture slot 3.



Desarrollo de videojuegos para Nintendo DS

Texto

```
consoleInit(&topScreen, 3, BgType_Text4bpp, BgSize_T_256x256, 31, 0, true, true);
```

Inicializa el modo texto para la consola superior o inferior; dependiendo del tamaño y de la configuración de vídeo, necesitará unos parámetros u otros.

```
consoleSelect(&topScreen);
```

Selecciona la consola a utilizar por *iprintf*; deben estar inicializadas con `consoleInit`.



Desarrollo de videojuegos para Nintendo DS

Texto

con `iprintf`, podemos escribir usando caracteres de escape para definir el color:

```
iprintf("\t\x1b[31mTest texto Superior\n");
```

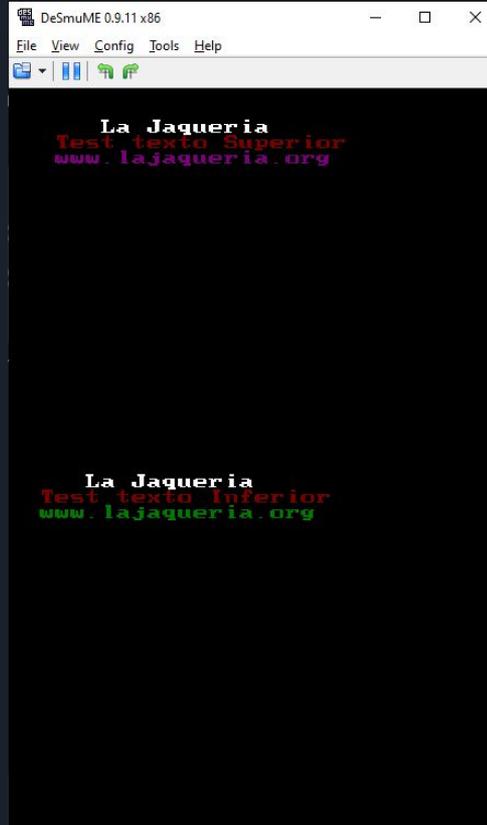
Tiene la posibilidad de 8 colores, escribiendo al principio `\x1b[nm` donde `n` es el número de color; pudiendo ser:

- **Negro** 30
- **Rojo Brillante** 31
- **Verde** 32
- **Amarillo** 33
- **Azul** 34
- **Magenta** 35
- **Cyan** 36
- **Blanco** 37

Además permite pintar definiendo la fila y columna usando el código `\x1b[x;yH` donde `x` e `y` son el nº de fila y columna.

Desarrollo de videojuegos para Nintendo DS

Texto





Desarrollo de videojuegos para Nintendo DS

Fondos

- Nintendo DS, permite definir por cada pantalla, 4 fondos simultáneos. Dependiendo de la configuración elegida, podremos tener distintas configuraciones.
- Cada fondo tiene una resolución en pantalla de 256x192 píxeles; compuesto por 768 tiles de 8x8 píxeles; sin embargo, se puede cargar un fondo mayor de hasta 511 x 255 píxeles.
- Nintendo DS, permite cargar imágenes de hasta 16bpp; es decir que permite tener hasta 65.536 colores en pantalla.



Desarrollo de videojuegos para Nintendo DS

Fondos

- Para poder cargar una imagen para nuestro juego, usaremos la utilidad que trae DevkitPro. Esta utilidad se llama GRIT y permite generar todos los ficheros necesarios para agregar a nuestro programa.
- Esta utilidad, permite a partir de una serie de parámetros importar los recursos a nuestro juego; ya sean fondos, mapas, paletas, sprites, etc....
- Para más información acerca de GRIT, os dejamos la dirección de su repositorio:

<https://github.com/devkitPro/grit/blob/master/grit-readme.txt>



Desarrollo de videojuegos para Nintendo DS

Fondos

Ejemplo de configuración para GRIT

```
-W3  
# disable alpha and set opaque bit for all pixels  
-gT!  
  
# use lz77 compression  
-gzl  
  
# 16 bit bitmap  
-gB16  
  
-gb
```



Desarrollo de videojuegos para Nintendo DS

Fondos

```
#include "background.h"

int main(void)
{
    videoSetMode(MODE_5_2D);
    videoSetModeSub(MODE_0_2D);
    consoleDemoInit();

    PrintConsole bottomScreen;
    consoleInit(&bottomScreen, 3, BgType_Text4bpp, BgSize_T_256x256, 31, 0, false, true);
    vramSetBankA(VRAM_A_MAIN_BG);

    // set up our bitmap background
    bgInit(3, BgType_Bmp16, BgSize_B16_256x256, 0, 0);

    decompress(backgroundBitmap, BG_GFX, LZ77Vram);
}
```



Desarrollo de videojuegos para Nintendo DS

Fondos

Veamos las funciones más importantes:

```
videoSetMode(MODE_5_2D);
```

Es importante configurar el modo de vídeo para usar los fondos.

```
vramSetBankA(VRAM_A_MAIN_BG);
```

También configurar el banco de memoria, para almacenar el fondo. En este caso para la pantalla de arriba.

```
bgInit(3, BgType_Bmp16, BgSize_B16_256x256, 0, 0);
```

Inicializamos el fondo número 3, para almacenar una imagen de 16 bits y de tamaño 256x256.

```
decompress(backgroundBitmap, BG_GFX, LZ77Vram);
```

Por último descomprimos la imagen en la posición de memoria designada para los fondos.

Desarrollo de videojuegos para Nintendo DS

Fondos



Desarrollo de videojuegos para Nintendo DS

Sprites

- Para Nintendo DS, se pueden utilizar hasta 256 sprites de distintos tamaños.
- Esto es ampliable gracias al uso de rotaciones, traslaciones y escalados conjuntamente.
- Se pueden presentar en 3 modos de color:
 - 16 paletas de 16 colores cada una (modo GBA).
 - 16 paletas de 256 colores.
 - Sprites con gráficos de 16bpp.
- Se pueden definir distintos tamaños de sprites.

8x8	16x8	32x8	
8x16	16x16	32x16	
8x32	16x32	32x32	64x32
		32x64	64x64

Desarrollo de videojuegos para Nintendo DS

Sprites

- A la hora de importar los sprites, se puede utilizar Grit.

```
-m!  
-gB8  
-gT 0F9C02  
#metatile  
-Mh4  
-Mw4
```





Desarrollo de videojuegos para Nintendo DS

Sprites

```
void animSprite(Sprite* sprite){
    int frame = sprite->anim_frame + sprite->state * FRAMES_PER_ANIMATION;
    u8* offset = sprite->frame_gfx + frame * 32*32;
    dmaCopy(offset, sprite->sprite_gfx_frame, 32*32);
}

void initSprite(Sprite * sprite, u8* gfx){

    sprite->sprite_gfx_frame = oamAllocateGfx(&oamMain, SpriteSize_32x32, SpriteColorFormat_256Color);

    sprite->frame_gfx = (u8*)gfx;
}
```



Desarrollo de videojuegos para Nintendo DS

Sprites

Veamos las funciones más importantes del ejemplo:

```
vramSetBankA(VRAM_A_MAIN_BG);  
vramSetBankB(VRAM_B_MAIN_SPRITE);
```

Configuramos el Banco A para almacenar el fondo y el Banco B de memoria para almacenar el Sprite.

```
oamInit(&oamMain, SpriteMapping_1D_128, false);
```

Se inicializa el motor de sprites para la pantalla principal.

```
initSprite(&elli, (u8*)ellireadyTiles);  
dmaCopy(ellireadyPal, SPRITE_PALETTE, 512);
```

Se inicializa el Sprite, y se copia la paleta usando DMA.



Desarrollo de videojuegos para Nintendo DS

Sprites

```
animSprite(&elli);  
oamSet(&oamMain, 0, elli.x, elli.y, 0, 0, SpriteSize_32x32, SpriteColorFormat_256Color,  
       elli.sprite_gfx_frame, -1, false, false, false, false);
```

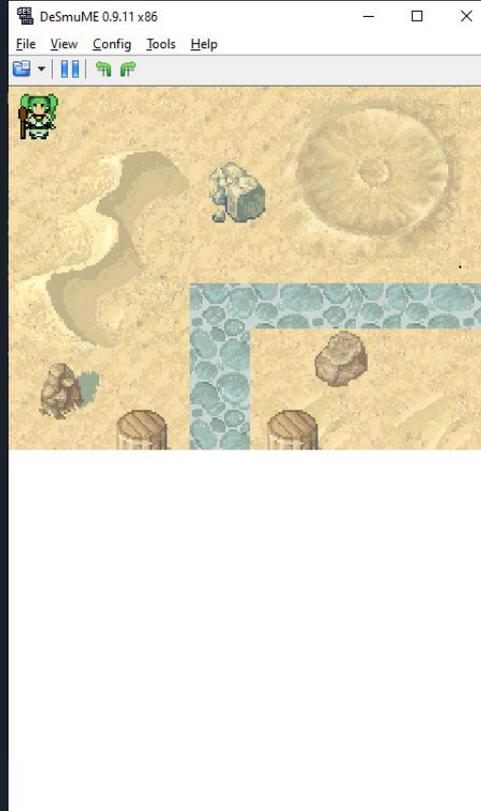
Se calcula la animación del sprite, y se muestra por pantalla, realizando la traslación y los gráficos a mostrar.

```
oamUpdate(&oamMain);
```

Se actualiza el motor de sprites.

Desarrollo de videojuegos para Nintendo DS

Sprites





Referencias

- https://es.wikipedia.org/wiki/Nintendo_DS
- <https://www.copetti.org/writings/consoles/nintendo-ds/>
- <https://devkitpro.org/>
- <https://libnds.devkitpro.org/>
- <https://github.com/devkitPro/grit>
- <https://riunet.upv.es/bitstream/handle/10251/11221/memoria.pdf?sequence=1>