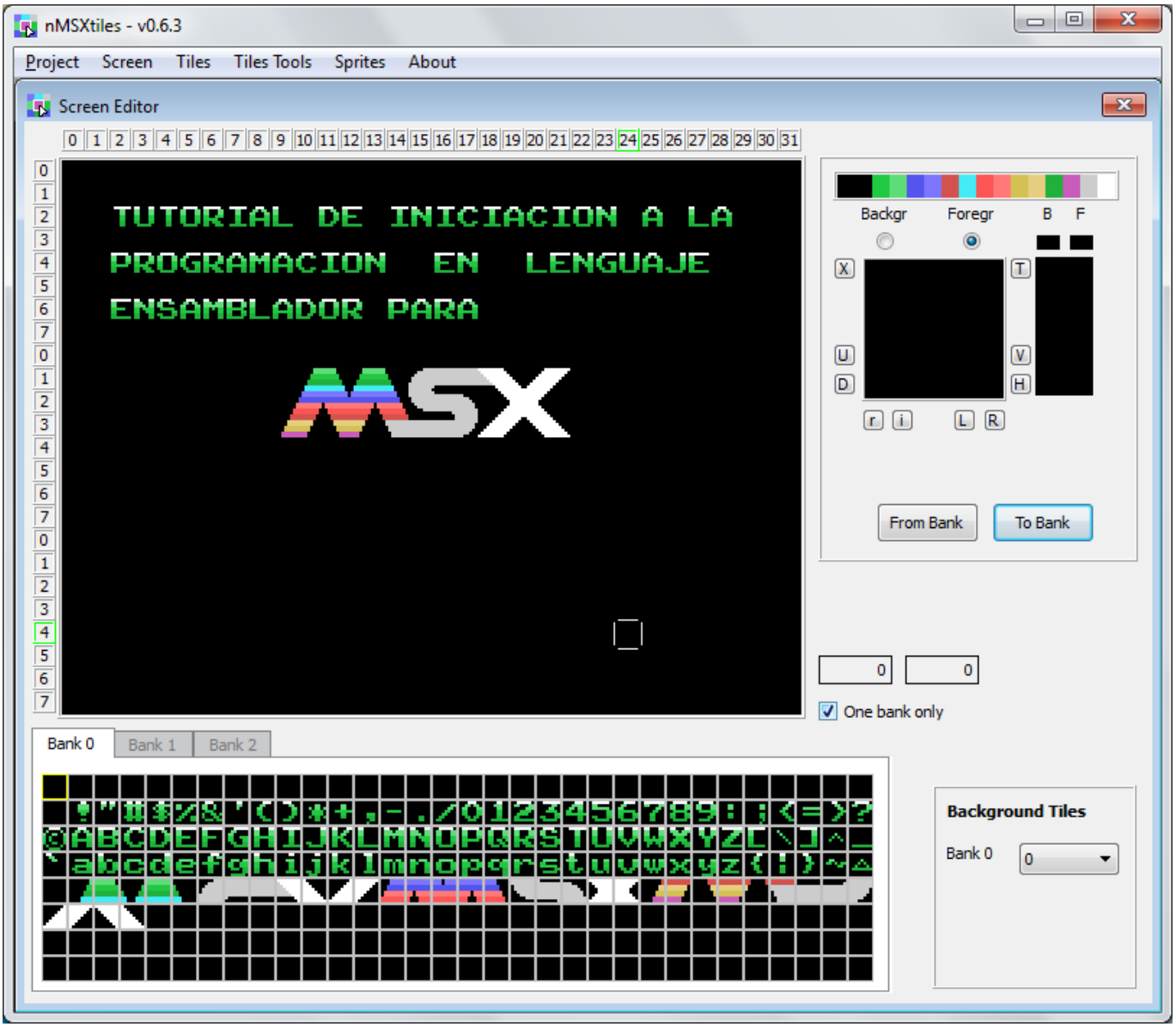
TUTORIAL DE INICIACION A LA PROGRAMACION EN LENGUAJE ENSAMBLADOR PARA MSX

5ª PARTE – EL MUNDO DE LOS GRAFICOS y 3

En esta parte del tutorial vamos a cambiar la forma de trabajar para mostrar el hola mundo grafico de otra manera que emplea menos código y bytes, tengo que decir que esta es la forma que yo uso para crear los menús en mi juego JumpinG. Veremos cómo importar nuestras imágenes o gráficos al nMSXtiles de Ramon de las Heras, que ha tenido a bien modificarlo con unas opciones que le he pedido. Veremos cómo se trabaja con este programa que empleo para todo el tema de menús y mapeados. Y daremos por finalizada la entrega sobre el Mundo de los Gráficos.



Esta será la herramienta principal para esta entrega del tutorial, el texto y el logo que veis dentro del nMSXtiles será el resultado final de nuestro tutorial, al que le llamare “HolaMundoGrafico4.asm” y que tenéis incluido en los ficheros suministrados para este tutorial aquí. Tutorial5.rar

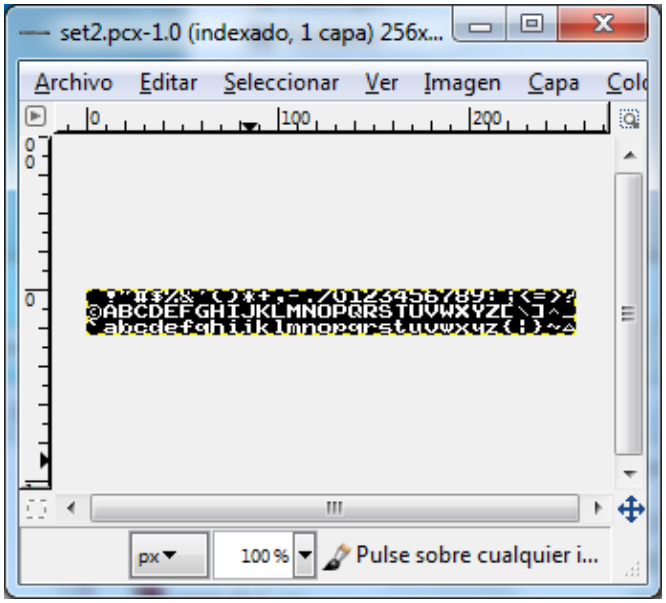
http://www.dimensionzgames.com/wp-content/uploads/downloads/2012/02/Tutorial5.rar

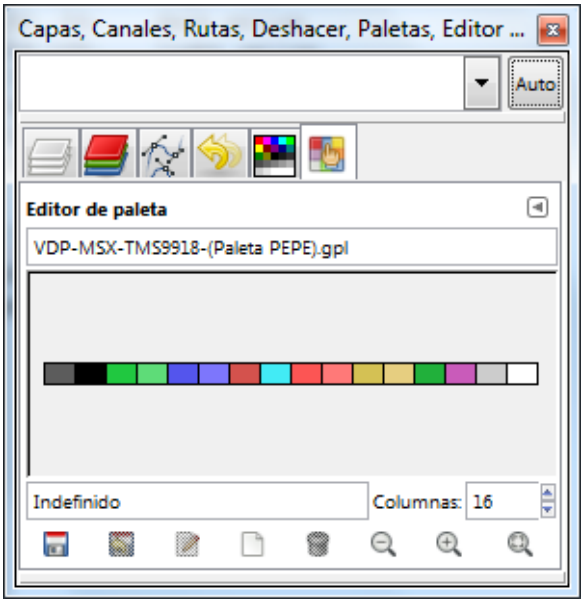
Lo primero que vamos a modificar es la paleta que usamos en GIMP, ya que el color transparente y el negro son el mismo y a la hora de importar nos puede hacer falta separar estos colores. Tienes que buscar dentro del directorio donde tienes instalado el GIMP hay una carpeta llamada palettes que es donde las almacena el GIMP o bien realizar una búsqueda en Windows del fichero de paleta que usamos VDP-MSX-TMS9918-(Paleta-PEPE).gpl puedes usar el EditPlus para modificarlo.

```
GIMP Palette
Name: VDP-MSX-TMS9918-(Paleta PEPE).gpl
Columns: 16
#
  0   0   0    Transparent
  0   0   0    black
 32 200  64    medium green
 94 220 120    light green
 84  85 237    dark blue
125 118 252    light blue
212  82  77    dark red
 66 235 245    cyan
252  85  84    medium red
255 121 120    light red
212 193  84    dark yellow
230 206 128    light yellow
 33 176  59    dark green
201  91 186    magenta
204 204 204    gray
255 255 255    Sin nombre
```

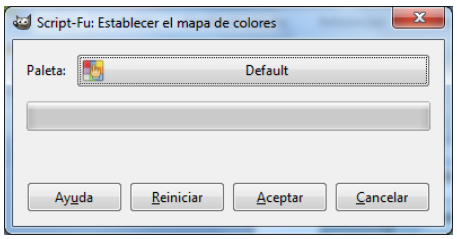
```
GIMP Palette
Name: VDP-MSX-TMS9918-(Paleta PEPE).gpl
Columns: 16
#
 92  92  92    Transparent
  0   0   0    black
 32 200  64    medium green
 94 220 120    light green
 84  85 237    dark blue
125 118 252    light blue
212  82  77    dark red
 66 235 245    cyan
252  85  84    medium red
255 121 120    light red
212 193  84    dark yellow
230 206 128    light yellow
 33 176  59    dark green
201  91 186    magenta
204 204 204    gray
255 255 255    Sin nombre
```

Aquí puedes ver que modificamos los tres 0 por tres 92 el color Transparent, de esta manera el negro será negro mientras que el transparente será gris oscuro. Guarda o graba el fichero.
Ahora vamos a abrir el set de caracteres con el GIMP fichero set2.pcx



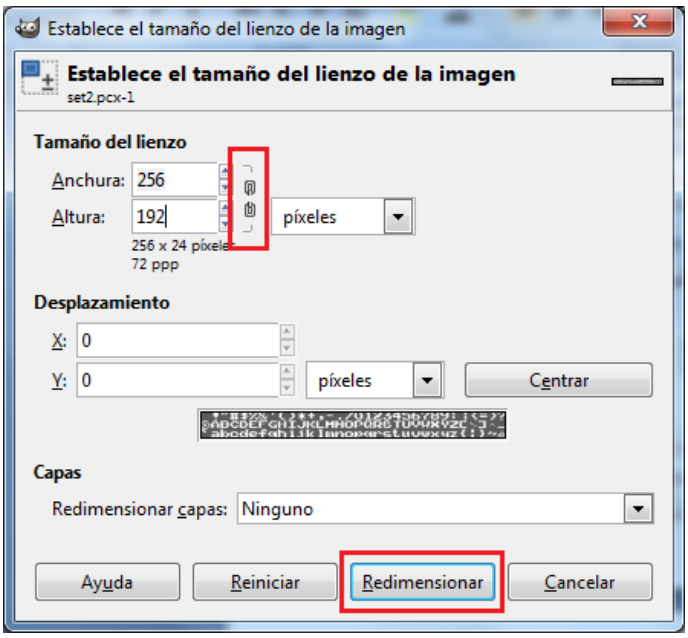


Aquí puedes ver la imagen PCX con nuestro set de caracteres y fíjate que en la paleta ahora tenemos el color Transparente en un gris oscuro.



Lo primero siempre después de abrir una imagen es cambiar la paleta del GIMP en modo DEFAULT por la del MSX Pulsa en los menús del GIMP en COLORES – MAPA – Establecer el mapa de colores… veras la imagen a la derecha de este texto.
Pulsa dentro de Default y en la ventana que se abre selecciona la paleta VDP-MSX-TMS9918-(Paleta-PEPE). Pulsa botón Cerrar.
Donde antes ponía Default ahora pone VDP-MSX-TMS9918-(Paleta-PEPE). Pulsa botón Aceptar.

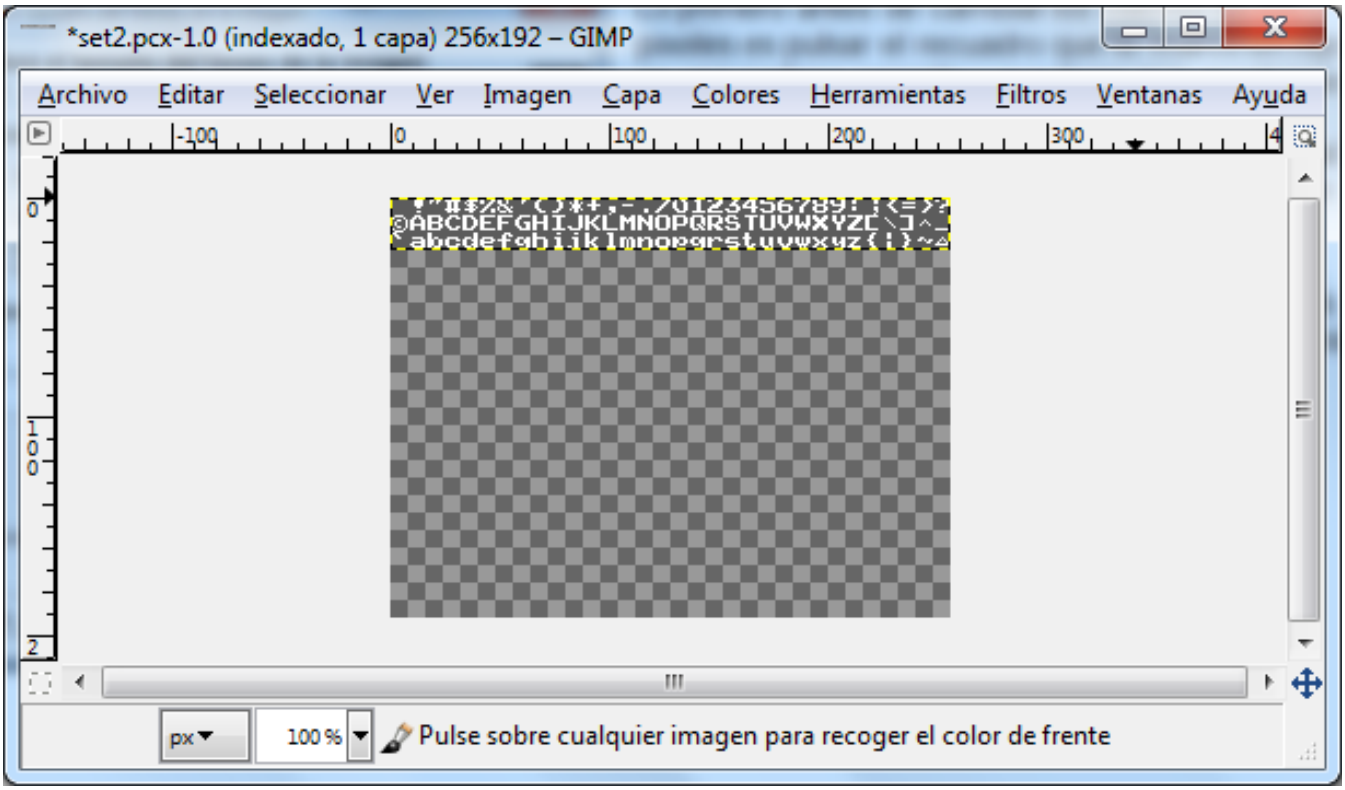
Ahora vamos a transformar el tamaño de la imagen a 256x192 ya que el importador del nMSXtiles no permite importar tamaños menores de esta resolución. Esto se realiza en los menús del GIMP en IMAGEN-Tamaño del lienzo… (Esto no altera la imagen, solo cambia el tamaño.)



Lo primero antes de cambiar los valores de los pixeles es pulsar el recuadro que te marco en rojo, es una cadena cerrada y que tiene que mostrarse como ves en esta imagen abierta, ya que esta opción es para que cree una imagen proporcional y nosotros lo que queremos es crear una imagen de un tamaño determinado.

Como nuestro ancho ya es 256 no lo cambiamos pero el alto que es 24 lo cambiamos a 192. Ahora pulsa en el botón Redimensionar. La ventana se cerrara, asegúrate que en redimensionar capas pone ninguno, sino se cierra esta ventana.
Con esto ya tendremos nuestra imagen en 256x192 pixeles que es la resolución del msx.

Fíjate que la imagen ha cambiado de tamaño pero sigue conservando intacto nuestro grafico.

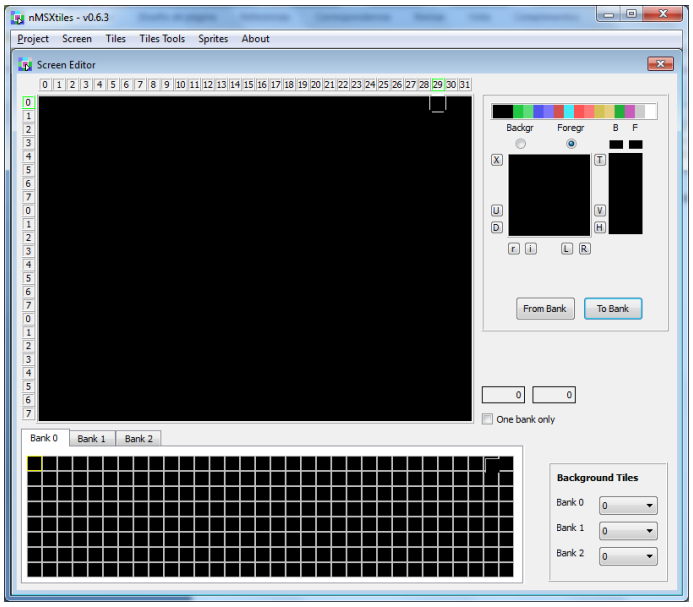


Ahora vamos a guardar la imagen en formato PNG para poderla importar con el nMSXtiles.
Pulsa en el menú Archivo del GIMP en la opción Guardar Como, en la ventana que se abre cambia el nombre set2.pcx a set2.png con esto le decimos que queremos grabarlo en PNG pulsa botón Guardar.

Veras que sale un mensaje de advertencia, ya que al cambiar el tamaño ha añadido una capa transparente y el PNG no puedes almacenar información de capas y otras cosas. Pulsa el botón Exportar y el solo se encargara de combinar las capas para grabar el PNG. En cuanto a los valores de exportación que salen en la otra ventana yo los dejo por defecto y pulso el botón Guardar

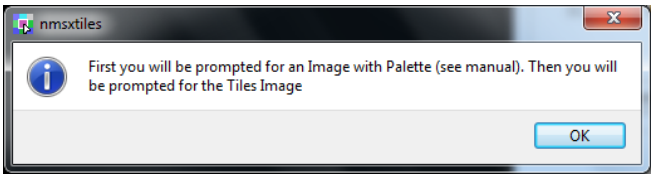
Ya estamos listos para importarlo al nMSXtiles, solo nos falta el fichero paletaMSX.png para que podamos importarlo, en el fichero TUTORIAL5.rar tenéis este fichero para utilizarlo.
Este PNG que he creado contiene un pixel de cada color de la paleta del MSX en el orden correcto de número de color, para que el nMSXtiles sepa a la hora de importar a que color pertenece cada tinta.

Ahora descomprime el nuevo nMSXtiles 0.6.4 que tienes comprimido en el fichero TUTORIAL5.rar con el nombre de nmsxtilesv0_6_4.zip en la carpeta C:\MSX donde tenemos el resto de herramientas.
Esta es una versión Beta y tiene menos estabilidad que la versión anterior debido a otras mejoras que el programador ha implementado para arrastrar los tiles desde los bancos al screen pero que de momento he podido comprobar que hay veces que se me cuelga el programa aunque puede ser porque uso Windows 7 64 Bits, aunque lo estoy utilizando porque es muy útil para extraer tiles y bancos y demás.

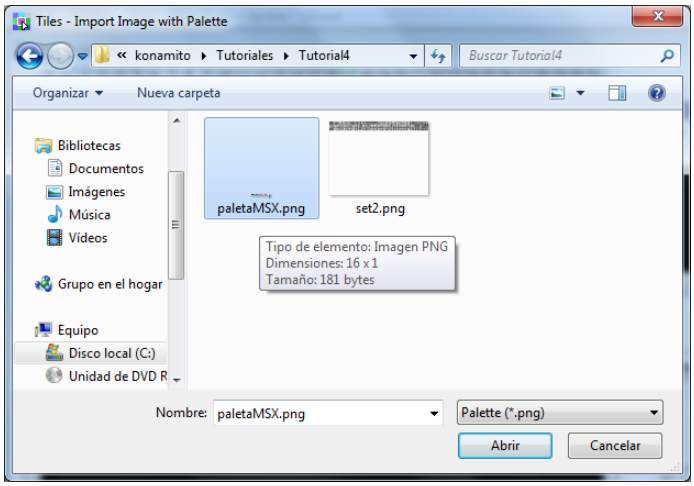


Ejecuta el nMSXtiles y vamos a crear un proyecto nuevo. En el menú pulsa Project – New veras que se muestra como en la imagen todo vacio.

Ahora vamos a importar la imagen PNG del set de caracteres creado con el GIMP.
Pulsa en el menú Tiles - Import

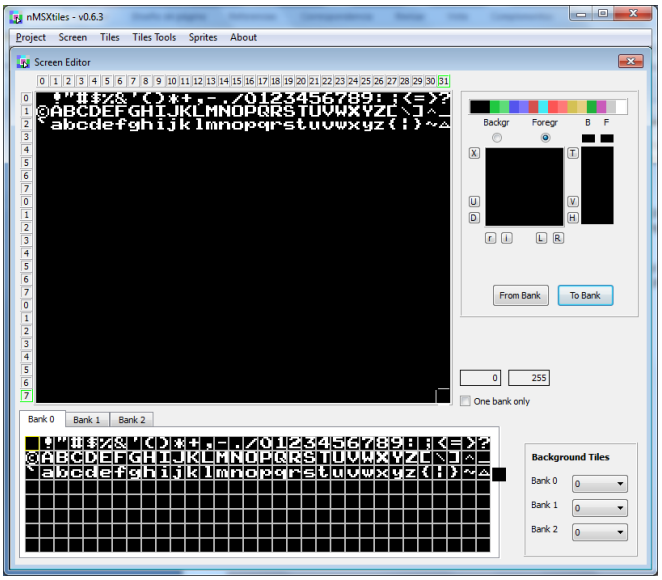


Veras que lo primero que te pide es el fichero con la paleta PNG Pulsa el botón OK.



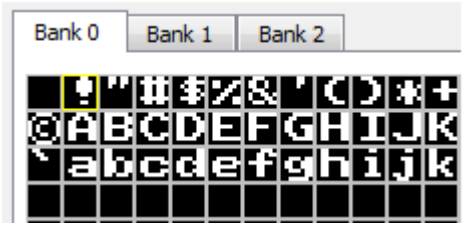
Busca el fichero paletaMSX.png
Suministrado con los ejemplos de este tutorial. Y pulsa el botón Abrir.

A continuación nos pide el fichero de imagen selecciona el fichero set2.png y pulsa de nuevo el botón Abrir.



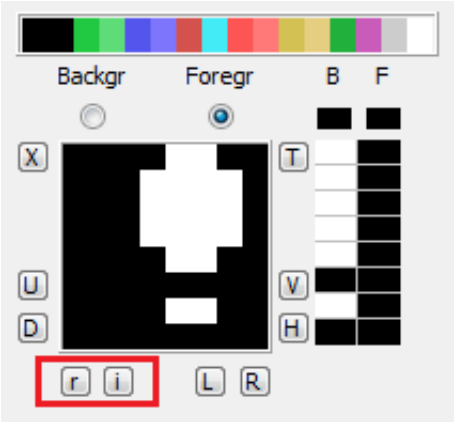
Y voila… Ya tenemos importado nuestro grafico creado con el GIMP o con el programa de gráficos que tu prefieras usar como photoshop, paint etc.

Ahora vamos a ver otros aspectos que hay que tener en cuenta a la hora de importar una imagen o grafico a nuestro proyecto en nMSXtiles.

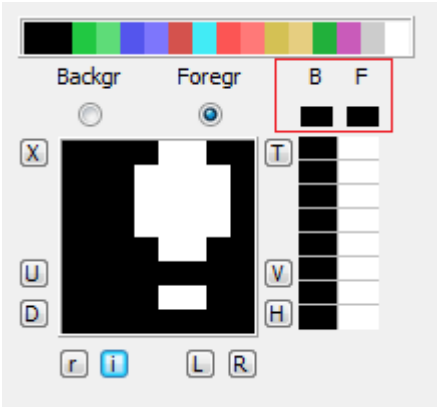


En la ventana de Bancos pulsa doble clic sobre el símbolo de admiración, con recuadro amarillo.

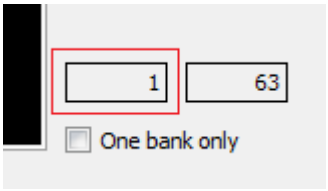
Todo esto te lo explico para que lo sepas para futuras importaciones de tus gráficos al nMSXtiles en blanco y negro.



Aquí en la ventana de edición del CHR puedes ver el símbolo de la admiración que tiene tinta blanca "B" para el fondo y tinta negra para las letras "F" invertido el color (porque ha pasado esto?) el importador ha visto que hay 2 tintas pero no sabe distinguir cual es el color del fondo y cuál es el color de la tinta cambiando el orden pero dejando el mismo resultado. Puedes dejarlo tal cual esta pero deberás tenerlo en cuenta a la hora de pintar en la pantalla. Yo lo modifico, así que pulso en el botón pequeño " i " para invertir el orden de los colores. Cuando la importación mezcla colores blanco en "B" y "F" puedes pulsar en el botón " r " para reordenar todos los colores a un lado y opcionalmente pulsar después en " i " para dejarlos en "B" o "F" Después deberás pulsar en el botón To Bank para llevar el carácter modificado al banco, y repetirlo para todos.

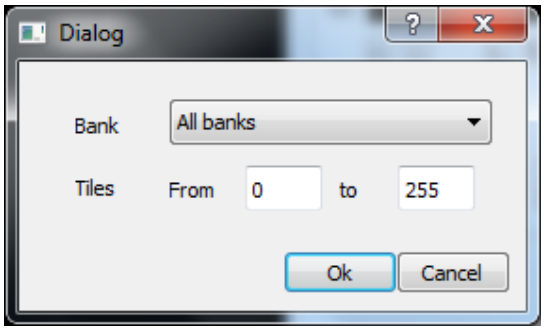


Puedes también mejorar otra cosa que la importación no tiene el cuenta si te fijas entre el cuerpo de la admiración y el punto no hay color blanco al igual que el ultimo byte del CHR, esto es normal porque al no haber pixeles no ha tenido en cuenta el color en esos bytes. Para que a la hora de comprimir no queden bytes en blanco y otros en negro y pierdas unos bytes en la compresión puedes pulsar en ese cuadrado negro justo debajo de la "F" que remarco en color rojo y pulsar a continuación en el color blanco, con esto le indicamos que todos los bytes de "B" o "F" sean de un mismo color. Pero ya que estas puesto a modificar esto después de estos pasos que te he explicado ya podrías poner un color especial a estas letras tu mismo. Te explico otro truco puedes poner una vez los colores en un CHR darle a CONTROL+C para que copie ese CHR, te sitúas en el siguiente CHR y con CONTROL+G puedes pegar solo el color ahorrándote tener que cambiar los 8 bytes en cada CHR. Así que tú mismo.



Vamos con una de la nuevas opciones agregadas a esta versión, extraer a fichero binario un número determinado de CHRs, para eso necesitamos saber el número de carácter inicial y el numero de carácter final, esto lo puedes ver en el nMSXtiles pulsando sobre el CHR y en la imagen que te pongo a la derecha de este texto. y que he remarcado en rojo. El primero es el CHR 0 y el ultimo es el CHR 95 con estos datos podemos ir a extraerlos.

Pulsa en el menú Tiles - Export bin data… En el cuadro de dialogo que se abre pon como nombre set2

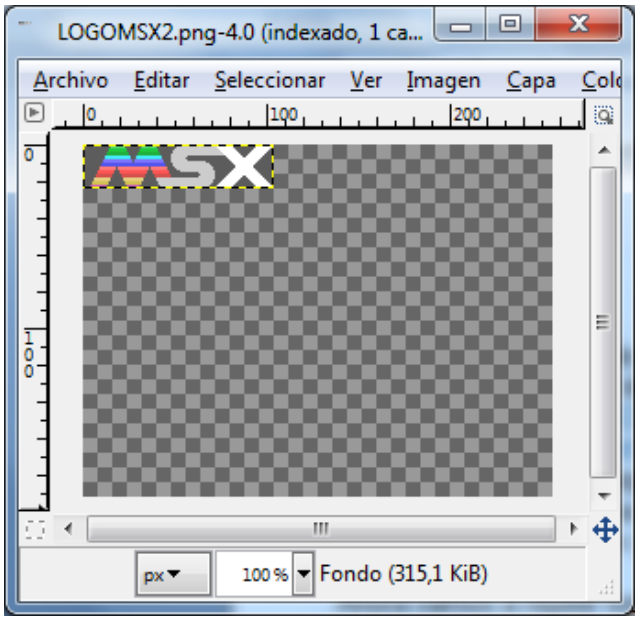


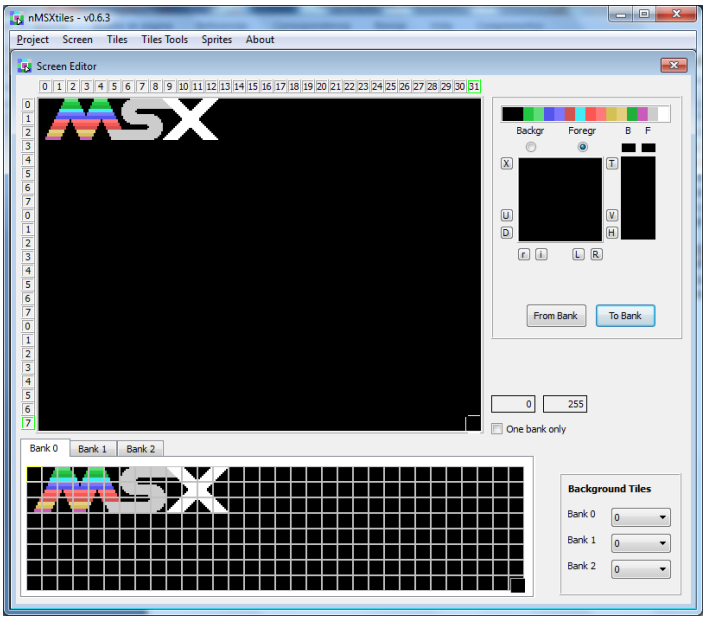
Veras que sale esta ventana emergente donde te deja seleccionar que numero de banco de los 3 que tenemos y de que numero de CHR a que numero de CHR quieres extraer.

Así que selecciono el Bank 0 y en Tiles selecciono From 0 to 95 y pulsamos el botón OK. Ahora nos ha guardado en el directorio 2 ficheros set2.til y set2.col que son el fichero de CHRs y los CLRs de los CHRs en formato binario. Podrías comprimirlos con Pletter como te he enseñado, Y convertirlos con binDB.exe a formato texto en ASM, Aunque no hace falta que lo hagas.

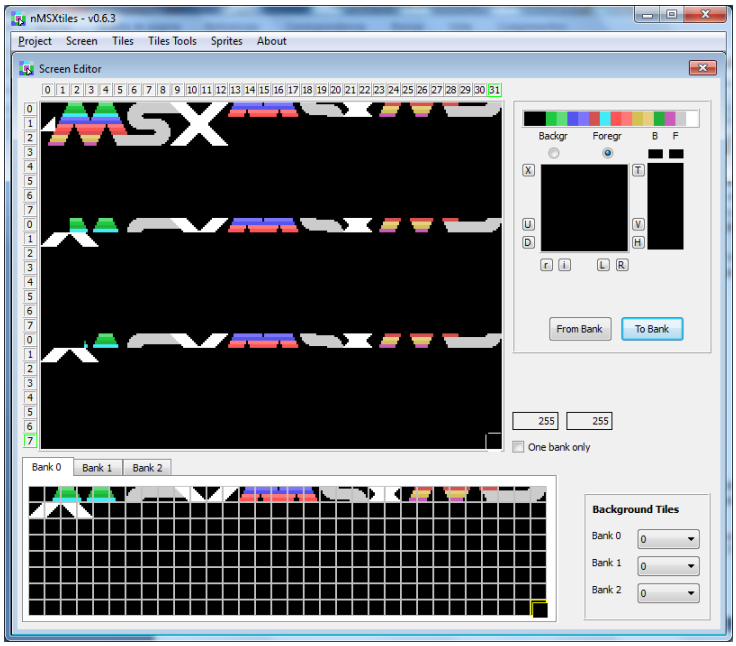
Sigamos explicando más cosas sobre el nMSXtiles
Ahora vamos a guardar el proyecto que se compone de 3 ficheros, uno el propio proyecto ".PRJ", otro el fichero de la pantalla ".SCR" y el ultimo el fichero con los 3 bancos de CHRs con sus colores. ".TIL"
Pulsa en el menú Project – Save lo primero que nos pide es el nombre de la pantalla, te voy a dar un consejo, yo creo una carpeta con el nombre Proyecto_set2 y pongo el mismo nombre para los 3 ficheros, de esta manera se que esos ficheros pertenecen a ese proyecto. Así que pongo a los 3 set2 sin especificar extensión ya que la pone el propio programa, grabándolo en la carpeta que he creado.

Ahora vamos a repetir todo el proceso desde final de la página 2, para importar el logoMSX al nMSXtiles, ya que lo tienes abierto pulsa en Project - New y minimízalo, ahora abre el GIMP con el fichero LOGOMSX2.PCX y sigue los mismos pasos que desde el final de la pagina 2. Que son asegurarte que esta seleccionada la paleta del msx, cambiar el tamaño del lienzo, grabarlo en formato PNG unificando las capas y listo. Ahora maximiza el nMSXtiles que dejaste minimizado.





Y realiza los mismos pasos como te enseñe a partir de la pagina 3, Menú Tiles - Import, selecciona el fichero con la paleta PNG, y a continuación la imagen del logoMSX2.png, una vez que la veas cómo se muestra en las capturas que te pongo encima, asegúrate que el orden del color está bien o puedes dejarlo tal cual que no pasa nada, el resultado en pantalla será el mismo sin cambiar colores.

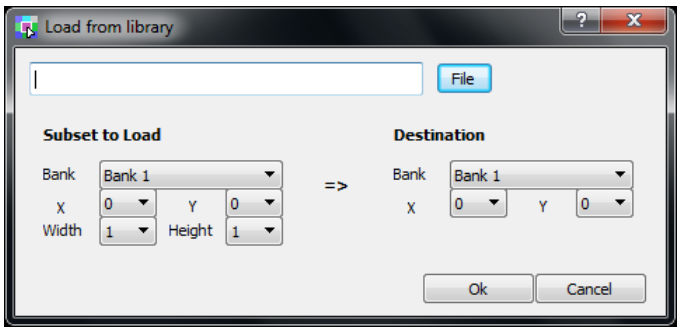


Ahora vamos a mover o poner todos los caracteres juntos o seguidos, ya que tenerlos de esta forma no es lo normal para poder exportarlos de tal número a tal número. Puedes hacerlo de forma manual o hacerlo con una opción que tiene el programa. En el menú Tiles Tools - Group Tiles puedes ver en la imagen que los tiles se han agrupado, ahora vamos a grabar el proyecto, crea una carpeta Proyecto_logoMSX y graba los 3 ficheros con el nombre de logoMSX.

Vamos a ver otras opciones del nMSXtiles, así que crea un nuevo proyecto.

Ahora te voy a enseñar como importar tiles sueltos de otros bancos o ficheros, para esto tienes que saber de qué fichero quieres cogerlos y el numero de tiles que te interesa importar. Con esto vamos generar el banco de CHRs que utilizaremos después para el HolaMundoGrafico4.asm.

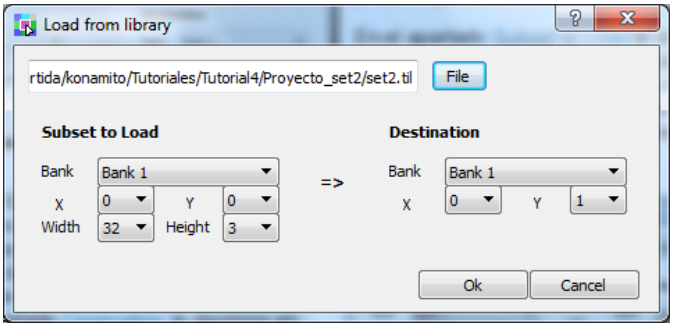
Pulsa en el menú del nMSXtiles en Tiles – Load from library… veras esta ventana emergente.



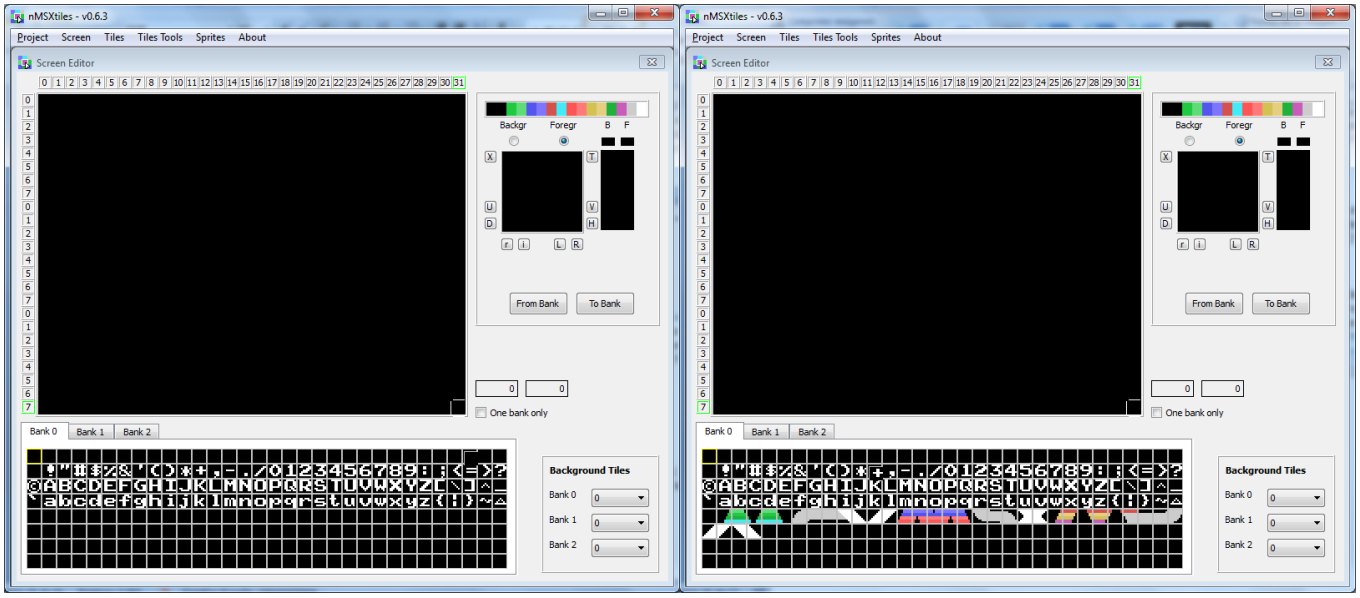
Pulsa en el botón File y selecciona el fichero set2.til que contiene los 3 bancos de tiles de la carpeta Proyecto_set2

En el apartado Subset to Load le indicamos del fichero que hemos cargado tres cosas.
1- de cuál de los 3 bancos vamos a leer, 2- en que CHR vamos a empezar diciendo la X y la Y, y
3- que ancho y alto de CHRs queremos leer.

Quiero leer del Banco 1 "Bank 1", "X=0 e Y=0" porque queremos empezar a leer desde el CHR 0, ancho le decimos "Width 32" que es el ancho del banco y alto le decimos "Height 3" que son las 3 filas que ocupan las letras en el banco.

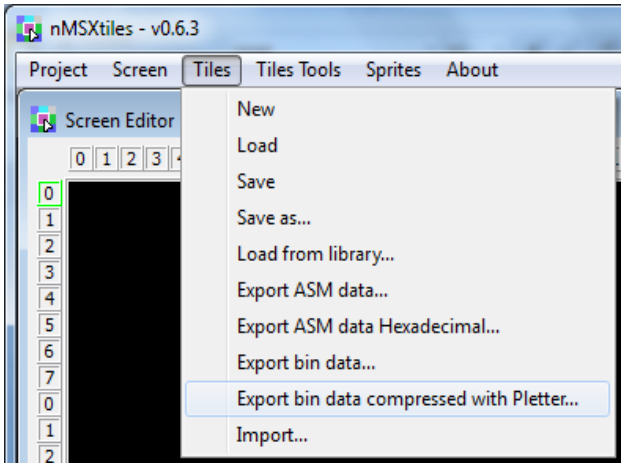


Ahora en el apartado Destination le decimos en que banco queremos dejarlo del proyecto actual que tenemos abierto, y que hemos creado nuevo por eso esta vacio. Aquí le digo Banco 1 "Bank 1" pero como quiero situar las letras del set de caracteres a partir del nº de CHR 32 le digo en "X=0 e Y=1" piensa que X e Y son la Columna y la Fila dentro del banco. Como el CHR 32 empieza en la columna 0, fila 1 debes saber también que el cuenta desde 0 a 31 la columna y de 0 a 7 la fila de cada banco. Por eso pongo X=0 e Y=1. Pulsa el botón OK. Veras la imagen debajo de este texto a la izquierda si lo has hecho bien.

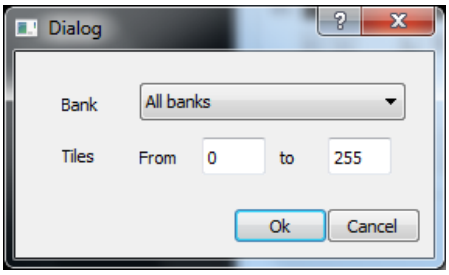


Ahora vamos a repetir el mismo proceso pero para importar los tiles de logo del MSX, supongo que ya sabrás hacerlo, te lo explico rápido menú Tiles-Load from library leo el fichero logoMSX.til de la carpeta Proyecto_logoMSX en el apartado Subset to Load pongo "Bank 1" y en "X = 0 e Y = 0" en "Width 32" y en "Height 2" en el apartado Destination pongo "Bank 1" en "X = 0 e Y = 4" y pulso el botón OK.
El resultado tiene que ser la imagen de la derecha encima de este texto si todo esta correcto.
Bien ya tenemos nuestro banco 0 preparado para nuestro nuevo ejemplo HolaMundoGrafico4.asm que veremos más adelante, pero antes nos queda por ver un par de cosas más del nMSXtiles.

El bank 0 está listo pero si pulsas en las pestañas del bank 1 y del bank 2 veras que están vacios, como lo que necesito es tener lo mismo en los 3 bancos, seguro que ya estas pensando en importarlo a los otros 2 bancos, pues no es así. Es más fácil de lo que piensas el propio nMSXtiles tiene un CheckBox llamado One bank only justo debajo de donde miras el numero de caracteres. Si marcas y desmarcas este CheckBox lo que hace es copiar el banco 0 sobre el banco 1 y el banco 2. De esta manera tendrías lo mismo en los 3 bancos puedes pulsar en los bancos para verlo, pero si estas creando menús es mas cómodo dejarlo puesto porque así no tendrás que estar cambiando de banco cuando pases de un tercio a otro de la pantalla. Lo siguiente es grabar los 3 ficheros del Proyecto con el nombre de tutorial5 en una nueva carpeta llamada Proyecto_tutorial5.



Otra cosa muy importante y muy cómoda es que podemos directamente exportar los bancos de CHRs o tiles a formato binario comprimido con Pletter directamente desde el nMSXtiles, Asi que voy al menú en Tiles - Export bin data compressed with Pletter... Guardo con el nombre de banco0.plet

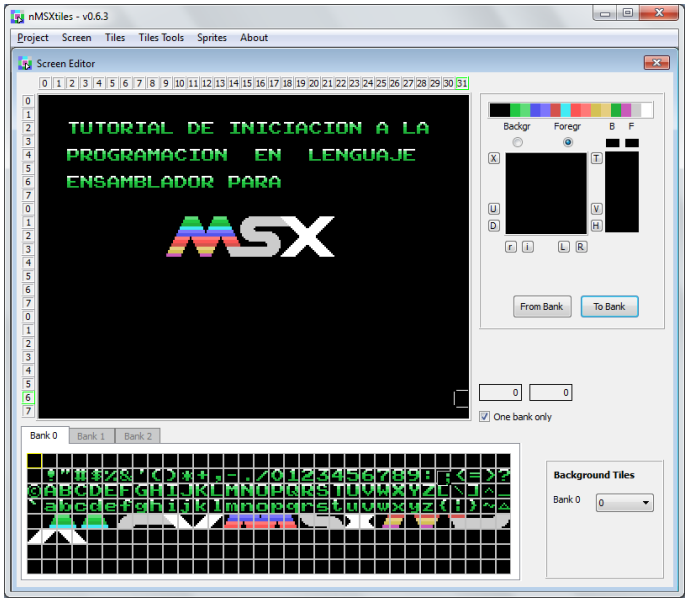


Selecciona Bank 0

y Tiles
From 0 to 255

Y botón OK

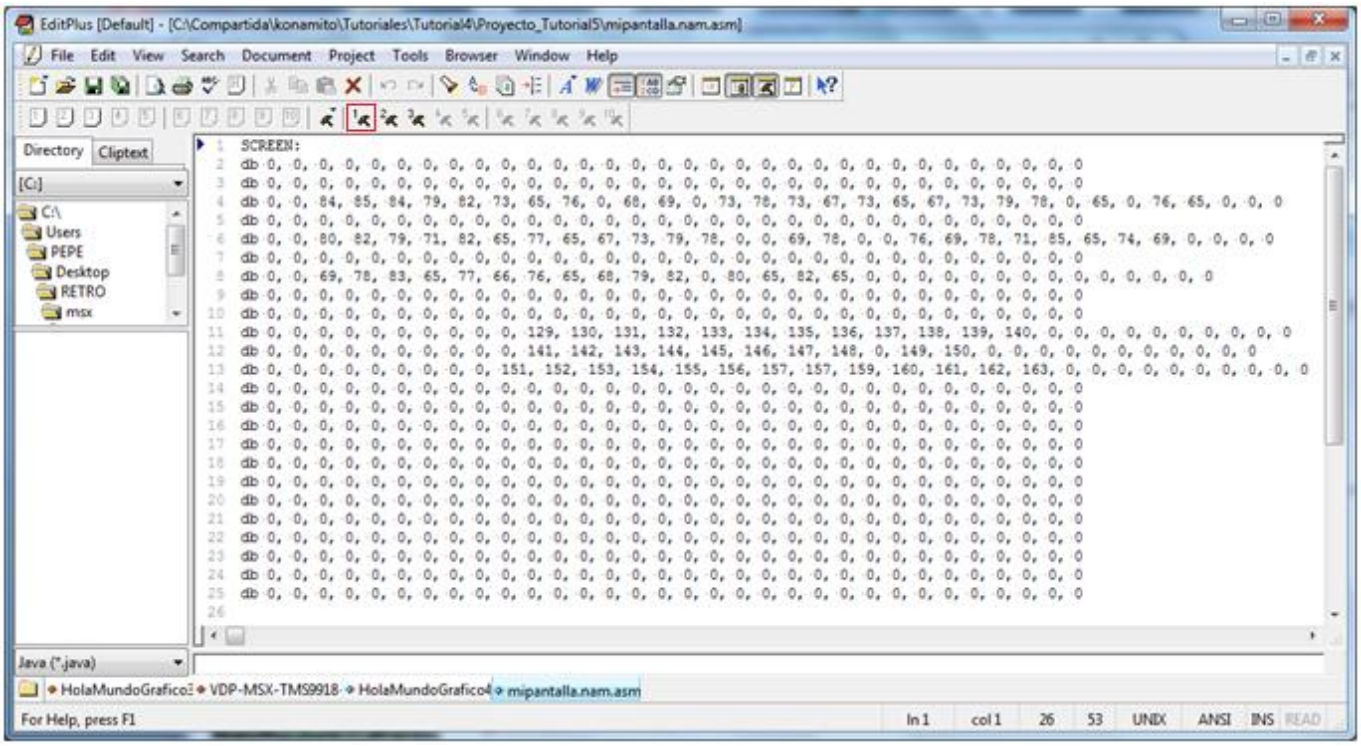
Ahora nos ha creado 2 fichero comprimido con Pletter con los CHRs y CLRs los fichero se llaman banco0.plet.til y banco0.plet.col y que tenemos ya listos para usar en nuestro código.



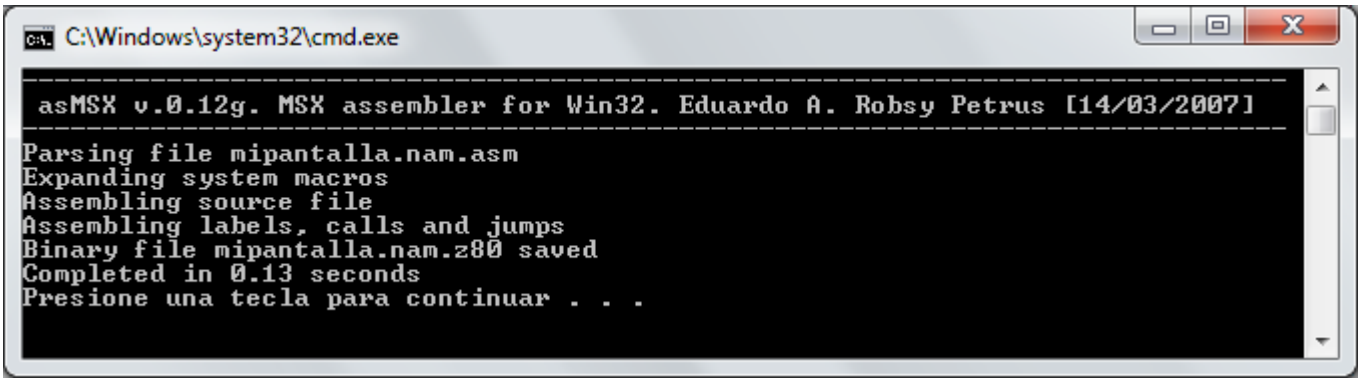
Ahora puedes ir colocando el texto y el logo del MSX o bien como lo pongo en esta imagen, o hazlo libremente a tu manera poniendo lo que quieras para que veas como es el proceso de creación de una pantalla.
Se trata de arrastrar o señalar el CHR que quieres usar del banco de CHRs y pulsar sobre la zona de la pantalla para situarlo en esa posición.

Una vez que tengas creada la pantalla vamos a grabarla comprimida en formato Pletter, pare eso pulsa en menú Screen – Export bin data compressend with Pletter... En el cuadro de dialogo para salvar el fichero guardalo dentro del mismo proyecto del tutorial5 y dale el nombre mipantalla.nam.plet de esta manera sabes que es mipantalla el ".nam" para que sepamos que es para la NAMTBL y ".plet" para indicar que esta comprimida con pletter. En mi caso la pantalla ocupa 768 Bytes, pero comprimida con Pletter se ha quedado en 129 Bytes, menos que todo el código que haría falta para crearla entre texto y código. Y sino fíjate en el código nuevo cuando creemos el HolaMundoGrafico4.asm

Ahora ya tenemos todo lo que nos hace falta para crear el HolaMundoGrafico4.asm. Pero voy a explicarte otra cosa más que puede ser interesante para más adelante. Igual que puedes grabar la pantalla en binario o binario comprimido con Pletter, también puedes grabar la pantalla en formato ASM en DB´s, esto puede ser útil si quieres agregarla al código, o bien cuando no quieres usar toda la pantalla completa porque tienes un marcador en un tercio o zonas de la pantalla que no usas. Para grabarla en formato DB´s solo tienes que ir al menú Screen – Export ASM data… y en el cuadro de dialogo que se abre pones mipantalla.nam.asm y la grabas en el directorio del proyecto. Ahora vete a ese directorio y abre el fichero con el EditPlus, primera imagen de la siguiente pagina, donde puedes ver los 768 Bytes compuestos de 32 chrs de ancho x 24 CHRs largo de alto.



Como puedes ver es fácil eliminar líneas enteras de esta pantalla ya que es texto modificable. Esto lo utilizo en mi juego JumpinG para eliminar las 2 primeras líneas que es donde tengo mi marcador, además de añadir aquí las tablas de movimientos de los enemigos. Pero volvamos a lo que te explicaba una vez que has eliminado las líneas que no te interesan y la etiqueta SCREEN: cómo puedo volver a pasar a binario este fichero, pues muy fácil, pulsa el botón compilar en EditPlus remarcado en rojo.



Aquí puedes ver que el asMSX ha generado un fichero binario llamado mipantalla.nam.z80 que podrías comprimir con Pletter y agregar a tu código, con un .incbin "NombreDelFichero"

Cierra el nMSXtiles y vamos con el EditPlus 3 crea un nuevo fichero y pega dentro este código.

```
;-----------------------------------------------------
; Nombre de nuestro programa
; Hola Mundo Grafico 02/02/2012
; Versión 4
;-----------------------------------------------------

;-----------------------------------------------------
; CONTANTES
;-----------------------------------------------------
; No definimos ninguna constante

;-----------------------------------------------------
; VARIABLES DEL SISTEMA
;-----------------------------------------------------
; Direcciones de la VRAM
        CHRTBL          equ     0000h   ; Tabla de caracteres
        NAMTBL          equ     1800h   ; Tabla de Nombres
        CLRTBL          equ     2000h   ; Tabla del color de los caracteres
        SPRATR          equ     1B00h   ; Tabla de los atributos de los sprites
        SPRTBL          equ     3800h   ; Tabla de Sprites
```

```
; Variables del Sistema MSX
       CLIKSW          equ     $F3DB   ; Keyboard click sound
       FORCLR          equ     $F3E9   ; Foreground colour

;-------------------------------------------------------
; DIRECTIVAS PARA EL ENSAMBLADOR ( asMSX )
;-------------------------------------------------------
       .bios           ; Definir Nombres de las llamadas a la BIOS
       .page 2         ; Definir la dirección del código irá en 8000h
       .rom            ; esto es para indicar que crearemos una ROM
       .start INICIO   ; Inicio del Código de nuestro Programa
; Seguir la norma del Standard MSX para una ROM
       dw 0,0,0,0,0,0 ; 12 Bytes a 0

;-------------------------------------------------------
; INICIO DEL PROGRAMA
;-------------------------------------------------------
INICIO:
       call    INIT_MODE_SCx          ; iniciar el modo de pantalla x
       call    INIT_GRAFICOS          ; colocar los gráficos en VRAM

; Colocar la NAMTBL comprimida en VRAM
       ld      hl,MIPANTALLA          ; Origen bytes de la NAMTBL
       ld      de,NAMTBL              ; destino NAMTBL
       call    DEPLET                 ; descomprimir en VRAM

FIN:
       jp      FIN             ; esto es como 100 goto 100 para que se quede en un bucle sin fin.

;-------------------------------------------------------

;-------------------------------------------------------
; INICIALIZA EL MODO DE PANTALLA Y COLORES
;-------------------------------------------------------
; BASIC: COLOR 15,1,1
; Establecer los colores
INIT_MODE_SCx:
       ld      hl,FORCLR       ; Variable del Sistema
       ld      [hl],15         ; Color del primer plano 15=blanco
       inc     hl              ; FORCLR+1
       ld      [hl],1          ; Color de fondo 1=negro
       inc     hl              ; FORCLR+2
       ld      [hl],1          ; Color del borde 1=negro
; call INITXT           ; BIOS set SCREEN 0
; call INIT32           ; BIOS set SCREEN 1
       call    INIGRP          ; BIOS set SCREEN 2
; call INIMLT           ; BIOS set SCREEN 3
;
; SCREEN 0 : texto de 40 x 24 con 2 colores
; SCREEN 1 : texto de 32 x 24 con 16 colores
; SCREEN 2 : gráficos de 256 x 192 pixeles con 16 colores
; SCREEN 3 : gráficos de 64 x 48 pixeles con 16 colores

; Esto es para quitar el sonido que emite el msx cuando se pulsa una tecla
       xor     a                       ; ld a,0
       ld      [CLIKSW],a              ; Variable BIOS desactivar sonido teclas

; salir de la rutina INIT_MODE_SCx
       ret
;-------------------------------------------------------

;-------------------------------------------------------
; COLOCA LOS GRAFICOS EN LA VRAM
;-------------------------------------------------------
INIT_GRAFICOS:
; Esto lo realizamos para que no se vea nada en la pantalla
; Mientras colocamos los CHR y los Colores en la VRAM
       call    DISSCR                  ; BIOS deshabilitar la pantalla

; Borrar los 768 Bytes de la Name Table - NAMTBL en VRAM
       ld      hl,NAMTBL               ; Dirección origen en VRAM
       ld      bc,768                  ; nº de bytes a rellenar 32 columnas * 24 lineas=768 Bytes
       xor     a                       ; a=0 - Valor a rellenar
       call    FILVRM                  ; BIOS -Fill block of VRAM with data byte

; aquí vamos a colocar el banco de CHRs en los 3 tercios en VRAM en la CHRTBL
; tercio 1
       ld      hl,banco0_CHR           ; Origen bytes de los CHRs
       ld      de,CHRTBL               ; destino CHRTBL tercio 1
       call    DEPLET                  ; descomprimir en VRAM
```

```
; tercio 2
        ld      hl,banco0_CHR           ; Origen bytes de los CHRs
        ld      de,CHRTBL+2048          ; destino CHRTBL tercio 2
        call    DEPLET                  ; descomprimir en VRAM

; tercio 3
        ld      hl,banco0_CHR           ; Origen bytes de los CHRs
        ld      de,CHRTBL+4096          ; destino CHRTBL tercio 3
        call    DEPLET                  ; descomprimir en VRAM

; aqui vamos a colocar el banco de CLRs en los 3 tercios en VRAM en la CLRTBL
; tercio 1
        ld      hl,banco0_CLR           ; Origen bytes de los CLRs
        ld      de,CLRTBL               ; destino CLRTBL tercio 1
        call    DEPLET                  ; descomprimir en VRAM

; tercio 2
        ld      hl,banco0_CLR           ; Origen bytes de los CLRs
        ld      de,CLRTBL+2048          ; destino CLRTBL tercio 2
        call    DEPLET                  ; descomprimir en VRAM

; tercio 3
        ld      hl,banco0_CLR           ; Origen bytes de los CLRs
        ld      de,CLRTBL+4096          ; destino CLRTBL tercio 3
        call    DEPLET                  ; descomprimir en VRAM

; Esto lo realizamos para que se muestre de nuevo la pantalla.
        call    ENASCR                  ; BIOS habilitar la pantalla

; Salir de la rutina INIT_GRAFICOS
        ret
;--------------------------------------------------------

;--------------------------------------------------------
; Rutina descompresora de RAM/ROM a VRAM pletter v1.1
; XL2S Entertainment
;--------------------------------------------------------
        .INCLUDE        "DEPLET.asm"
;--------------------------------------------------------

;--------------------------------------------------------
; Banco de CHRs comprimido con Pletter
;--------------------------------------------------------
banco0_CHR:
        .INCBIN         "Proyecto_Tutorial5\banco0.plet.col"
;--------------------------------------------

;--------------------------------------------------------
; Banco de CLRs comprimido con Pletter
;--------------------------------------------------------
banco0_CLR:
        .INCBIN         " Proyecto_Tutorial5\banco0.plet.col"
;--------------------------------------------

;--------------------------------------------------------
; mi pantalla NAMTBL comprimida con Pletter
;--------------------------------------------------------
MIPANTALLA:
        .INCBIN         " Proyecto_Tutorial5\mipantalla.nam.plet"
;--------------------------------------------


;--------------------------------------------
; FINAL DE NUESTRO CODIGO EN ENSAMBLADOR.
;--------------------------------------------
```

A estas alturas del tutorial estoy seguro que ya debes saber que hace el HolaMundoGrafico4.asm, pero le daré un pequeño repaso para explicar las cosas nuevas, aunque con lo comentarios debería bastarte.

La cabecera del programa las variables y las direcciones VRAM están más que explicadas en anteriores tutoriales, creo que lo único que no he comentado es lo de seguir la normativa después de la directiva .start INICIO para que sea más compatible con la norma de una ROM hay que dejar 12 bytes a 0 hasta el comienzo de nuestro código, si esto lo quitas funcionara sin afectar a nada, pero siempre es mejor seguir las normativas del estándar.

La rutina INIT_MODE_SCx como siempre establece o pone el SCREEN2, añadiéndole que quite el sonido de la pulsación de las teclas. Más adelante veremos que añadiré más cosas a esta rutina.

La rutina INIT_GRAFICOS veras que ha cambiado muy poco de los otros ejemplos pero es fácilmente entendible, primero desactivo la pantalla para que no se vea nada mientras colocamos los CHRs y los CLRs, borro la NAMTBL, después descomprimimos los bytes del banco 0 que hemos creado con el nMSXtiles a la VRAM en cada uno de los 3 tercios, esto mismo lo repetimos con el color de los CHRs para finamente volver a activar la pantalla al haber finalizado de colocar gráficos en la VRAM.

Ahora en vez de colocar en la NAMTBL en la columna X fila Y diferentes textos, y colocar leyendo desde una tabla los CHRs de los gráficos del logoMSX en cada una de la líneas. Pasamos a descomprimir directamente la pantalla en la NAMTBL quitándonos toda esta labor.

```
; Colocar la NAMTBL comprimida en VRAM
        ld      hl,MIPANTALLA           ; Origen bytes de la NAMTBL
        ld      de,NAMTBL               ; destino NAMTBL
        call    DEPLET                  ; descomprimir en VRAM
```

Ahora te explico el ahorro de memoria que siempre es muy importante. Vamos a contar los bytes que ocupa en formato tutorial4 y el mismo ejemplo haciéndolo con el formato empleado en este tutorial.

Formato tutorial4:
Set de letras CHRs = 504 bytes, LogoMSX CHRs =153 bytes, LogoMSX CLRs = 144 bytes – TOTAL 801 Bytes
Formato nuevo:
Set de letras CHRs + LogoMSX CHRs = 623 bytes- set de letras CLRs + LogoMSX CLRs = 181 bytes – TOTAL 804 Bytes

Puedes ver que tenemos solo 3 bytes más, pero ya tenemos puesto el multi-color de las letras.

Formato tutorial4:
Código 86 bytes+ texto 18 bytes+ tabla color 8 bytes+ tabla nanlogo 39 Bytes+ rutina copyblock 20 Bytes = TOTAL 171 Bytes
Formato nuevo (como en el tutorial4, pantalla comprimida con el texto HOLA MUNDO Grafico y el logoMSX en 1 tercio)
Codigo 27 Bytes + Pantalla HolaMundo.nam.plet = 81 bytes – TOTAL 108 Bytes

Como puedes ver en el formato tutorial4 hemos empleado 972 Bytes, mientras que en este tutorial haciendo lo mismo hemos empleado 912 Bytes, hemos gastado 60 Bytes menos, además de facilitarnos mucho mas el trabajo reduciendo el código.

Seguimos con el ejemplo:

```
;-------------------------------------------------------------
; Rutina descompresora de RAM/ROM a VRAM pletter v1.1
; XL2S Entertainment
;-------------------------------------------------------------
        .INCLUDE        "DEPLET.asm"
;-------------------------------------------------------------
```

Aquí puedes ver que he quitado todo el código de la rutina descompresora, lo he grabado en un fichero nuevo al que le he dado el nombre DEPLET.asm. La directiva .INCLUDE “nombre_fichero” lo que hace es incluir el código en ensamblador en este mismo punto donde está la directiva, es como si hiciéramos un paste del código de la rutina descompresora en este punto compilando nuestro código mas el código de la rutina DEPLET.asm de esta manera podemos separar rutinas o partes de código que sabemos que funcionan bien, y no nos hace falta ver en el código general ocupándonos espacio, o bien porque en un momento determinado nos puede hacer falta cambiar el sistema de descompresión por otro sistema, y con solo cambiar esta rutina ya tendríamos modificado el código de nuestro programa.

```
;-------------------------------------------------------------
; Banco de CHRs comprimido con Pletter
;-------------------------------------------------------------
banco0_CHR:
        .INCBIN         "Proyecto_Tutorial5\banco0.plet.til"
;-----------------------------------------

;-------------------------------------------------------------
; Banco de CLRs comprimido con Pletter
;-------------------------------------------------------------
banco0_CLR:
        .INCBIN         "Proyecto_Tutorial5\banco0.plet.col"
;-----------------------------------------
```

Aquí puedes ver que he quitado del código los bytes de los gráficos comprimidos con Pletter, haciendo uso de la directiva .INCBIN “nombre_fichero” lo que hace en incluir un fichero binario en esta parte del código, funciona como la directiva .include pero en vez de añadir código .asm lo que añade son DB´s con datos, por eso antes de la directiva le añado una etiqueta banco0_CHR: o banco0_CLR: para después desde código saber en qué dirección empiezan esos bytes a los que quiero acceder.
Además de ahorrarme tener que pasarlos por el programa binDB y pegarlo en el código, también puedes ver que añado el directorio donde tiene que leer los fichero que están en Proyecto_Tutorial5

```
;-------------------------------------------------------
; mi pantalla NAMTBL comprimida con Pletter
;-------------------------------------------------------
MIPANTALLA:
        .INCBIN         " Proyecto_Tutorial5\mipantalla.nam.plet"
;-------------------------------------------
```

Y para finalizar también he INCluido un BINario con la pantalla que creamos con el nMSXtiles comprimida con Pletter al que le he añadido la etiqueta MIPANTALLA: para localizar la dirección.
De esta manera con esta parte del código sabe que tiene que coger los Bytes de los DB´s que hay en la posición de memoria que indica MIPANTALLA y descomprimirla en la NAMTBL.

```
; Colocar la NAMTBL comprimida en VRAM
        ld      hl,MIPANTALLA           ; Origen bytes de la NAMTBL
        ld      de,NAMTBL               ; destino NAMTBL
        call    DEPLET                  ; descomprimir en VRAM
```

Otra cosa que no he explicado en los demás tutoriales, es que no tenemos que indicar el número de bytes a transferir porque la rutina de descompresión Pletter tiene ya implementado en el código funciones para saber cuándo se terminan de descomprimir datos.

El resultado final será esta imagen o lo que vosotros hayas puesto.

Espero ver vuestros comentarios y vuestras capturas de pantallas, en los BLOG´s o FORO´s del tutorial.

Espero que haya sido de vuestro total agrado y nos vemos en el próximo tutorial. Donde explicare como se crean y como se manejan los sprites como incorporarlos al código fuente de nuestro programa. Así como los programas y herramientas que utilizo para este cometido también en varias partes.

José Vila Cuadrillero

"ES DETESTABLE ESA AVARICIA ESPIRITUAL QUE TIENEN, LOS QUE SABIENDO ALGO, NO PROCURAN LA TRANSMISION DE ESOS CONOCIMIENTOS."
Miguel de Unamuno
Escritor y Filósofo.
(Bilbao 1864 - Salamanca 1936)