UNIVERSITAT POLITÈCNICA DE VALÈNCIA

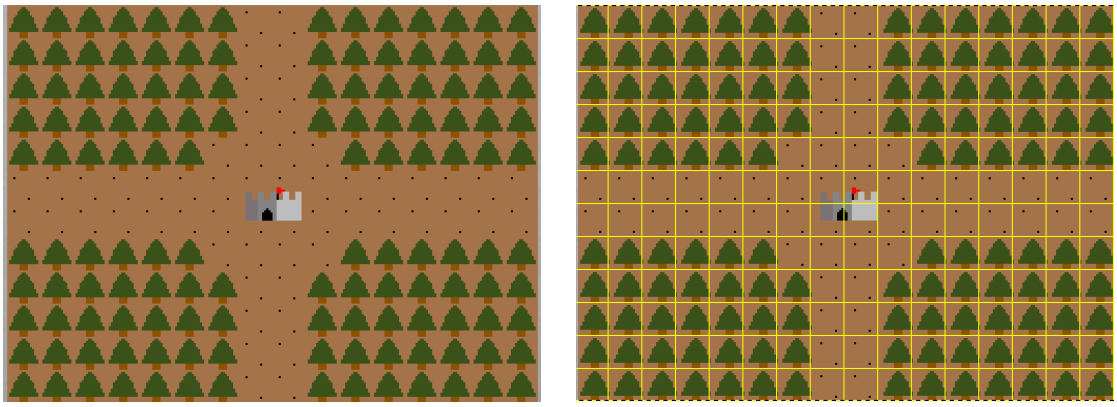
Generación de imágenes para escenarios teselados en videojuegos sobre NDS

Apellidos, nombre	**Agustí Melchor, Manuel** (magusti@disca.upv.es)
Departamento	**Dpto. de Ing. De Sistemas y Computadores (DISCA)**
Centro	**Escola Tècnica Superior d'Enginyeria Informàtica** Universitat Politècnica de València

1 Resumen de las ideas clave

En este artículo vamos a abordar cómo se puede crear una imagen compleja que va a ser utilizada como un escenario para un videojuego en la Nintendo DS (NDS). Habitualmente, estas imágenes no pueden ser estáticas, esto es, no son creadas completamente con anticipación, puesto que parte de su contenido depende del instante de ejecución, de las acciones del usuario. Además, por limitaciones del *hardware* o por optimizar recursos, hay partes de su contenido que se repiten, lo que ayuda a mantener bajo el consumo de recursos y ofrecer una experiencia de usuario fluida.

Así, por ejemplo, podemos ver situaciones como la que muestra la Figura 1: la imagen de la izquierda es un escenario y, a la derecha, se muestra la descomposición de esta en teselas: componentes cuadrados que se han ido ubicando uno al lado del otro hasta componer la imagen que ve el usuario, como si de ladrillos de una pared se tratara. En el caso de la Figura 1b, que muestra una rejilla superpuesta a la imagen, podemos observar que el número de teselas diferentes para componer la figura se reduce a seis: los árboles, el camino y el castillo (que está compuesto por cuatro teselas).



a) *b)*

Figura 1: Ejemplo de imagen para escenario: a) imagen resultante, b) descomposición en teselas.

Esta idea es utilizada actualmente en aplicaciones web y para móviles de mapas y cartografía (GIS)[1], que nos hemos acostumbrado a utilizar y que sustituyen en muchos casos al mapa o plano de papel. Esto se ha dado, por un lado, por la comodidad de llevarlos prácticamente siempre con nosotros en el teléfono móvil y, por otro, porque ofrecen con una cantidad relativamente baja de datos transferidos, la visualización de una imagen tremendamente grande. Si bien es cierto que no tenemos delante todo el mapa, como hacíamos con los tradicionales, a cambio, se nos ofrece nuestra ubicación dentro del mapa, y un área a nuestro alrededor mayor que la que podemos alcanzar con la velocidad y dirección de nuestro movimiento.

[1] Podemos encontrar más información al respecto en “Why tiled maps?”, disponible en <https://www.e-education.psu.edu/geog585/node/706>.

Para facilitar el seguimiento de este trabajo se han dispuesto todos los elementos del proyecto que aquí se comenta en *GitHub* [2].

2 Objetivos

Una vez que el lector haya leído con detenimiento este documento:

- Habrá visto los conceptos básicos de desarrollo de escenarios para videojuegos en la consola NDS, a partir de imágenes o texturas.
- Habrá visto el proceso de generación y transformación de los recursos de naturaleza gráfica que se incorporan a la aplicación.
- Dispondrá de un enlace a un repositorio [2], de donde obtener un ejemplo de código que permite generar un escenario a partir de un número mínimo de componentes gráficos en forma de imágenes en mapa de bits.

No es un objetivo describir el uso de emuladores, para ejecutar las aplicaciones para la NDS, como *DeSmuMe* [3], así como tampoco instalar las herramientas que permiten la creación del ejecutable o la carga del mismo en la consola, para ello se puede recurrir a los trabajos de [4] y [5]. Tampoco es un objetivo de este trabajo entrar a ver las características de la NDS. Si el lector tiene curiosidad o necesidad de profundizar en estas cuestiones, es recomendable consultar [6].

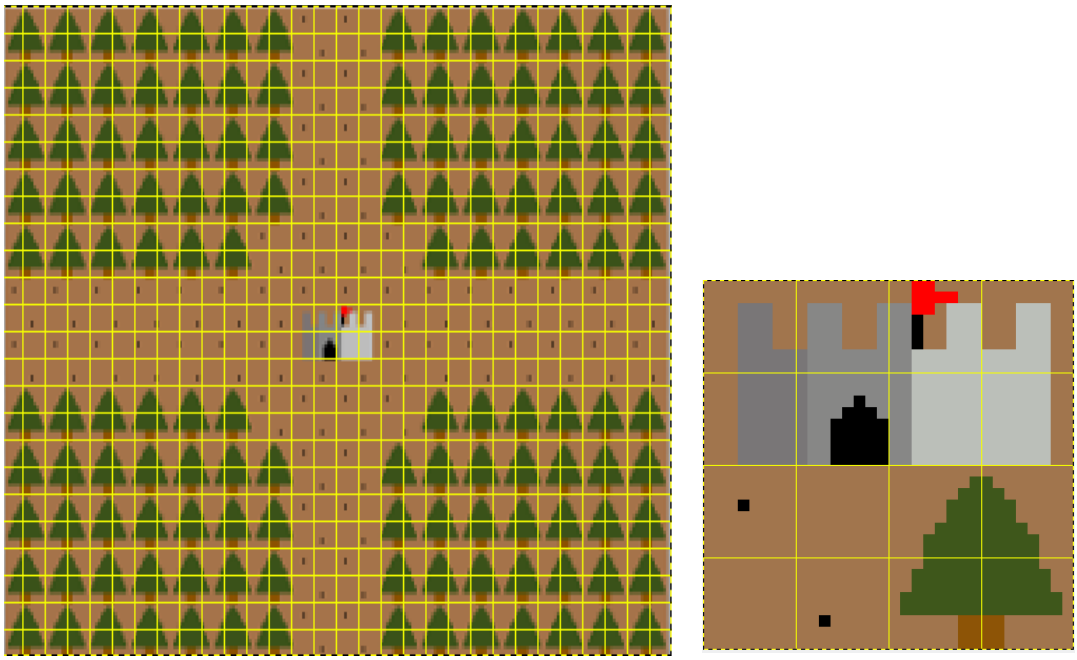
Para ver estos objetivos vamos a seguir el tutorial de *P. Rising* [1], su inacabada pero interesantísima guía de desarrollo de aplicaciones para la NDS. El ejemplo de código de fondos teselados no se puede aplicar, puesto que ha habido cambios en las librerías en que se basa. Así que, parte de nuestra aportación será actualizar el modo de proceder que se describe en este tutorial en cuanto a la manera de generar el mapa de caracteres, ya que no se dispone de la opción mencionada en la versión actual y no se puede hacer como lo describe el tutorial original.

3 Introducción

En el caso de la NDS, las casillas o teselas son de 8x8 píxeles (también se podría trabajar en 16x16, 32x32 y 64x64) y con una paleta de colores asociada. La resolución de la pantalla es de 240x192 píxeles, por lo que son necesarias 32 filas x 24 columnas de teselas para rellenar la pantalla. En la Figura 2a, se puede ver, sobre la imagen a la resolución nativa de la NDS, sobrepuesta en color amarillo, esta rejilla de teselas de 8x8 píxeles. Si nos fijamos, podemos observar que, con solo las 16 teselas que se muestran en la Figura 2b, se consigue reconstruir toda la imagen.

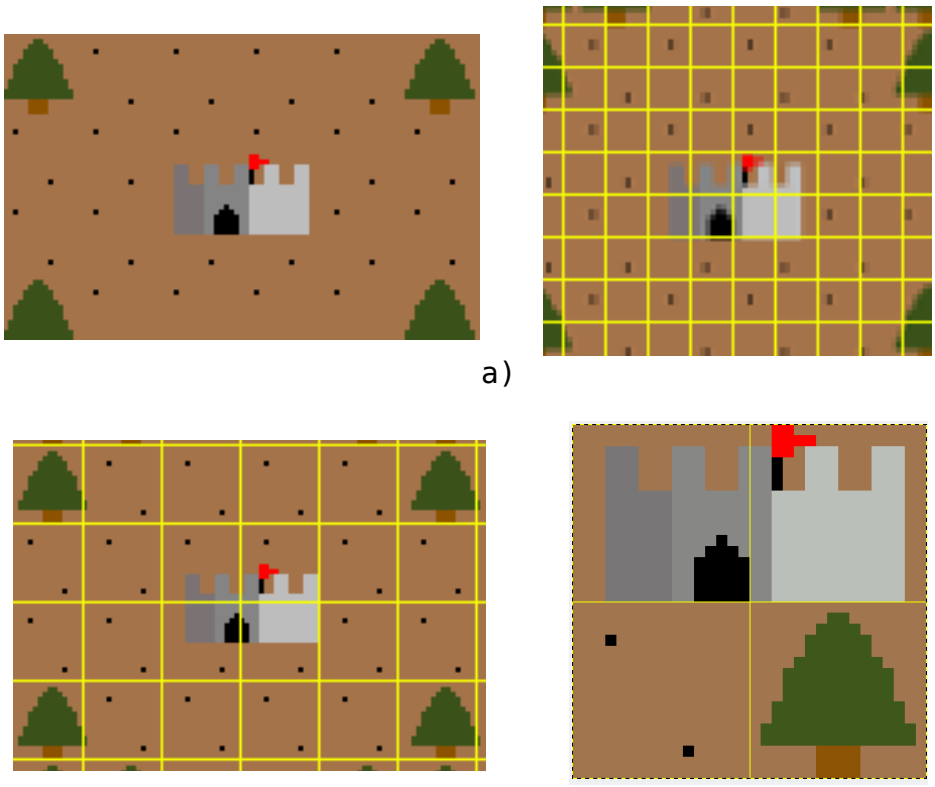
Otras particiones parecen posibles, pero si fijamos la atención en la zona central, Figura 3a, la rejilla de 8x8 muestra un conjunto de teselas compatible con el dibujo, esto es, que lo puede replicar. En cambio, la de 16x16, Figura 3b, observará el lector, que no podría replicar el castillo que está en el centro de la imagen, puesto que el “suelo” a su alrededor tiene unos puntos negros que no son compatibles (que no son realizables) con este conjunto de teselas, de 16x16, como generador.

La relación entre la imagen que contiene todas las teselas y la imagen a componer es lo que se denomina un **mapa**, una estructura de datos que describe en cada posible casilla de la imagen resultante, qué tesela del posible repertorio (catálogo) se utilizará.



a) b)

Figura 2: Imagen y conjunto de teselas generadoras para el caso de la NDS: (a) rejilla sobrepuesta y (b) catálogo de teselas de 8x8 píxeles.



a)

b)

Figura 3: Rejillas de 8x8 y de 16x16. Observe el detalle en las imágenes ampliadas de la zona central: (a) el motivo y la rejilla sobrepuesta de 8x8 píxeles, (b) rejilla sobrepuesta de 16x16 píxeles y conjunto de teselas que la generan.

La pantalla se puede ver entonces como formada por una rejilla (*grid*) o tablero, cuyas casillas (*screen blocks*) son rellenadas por las teselas (también denominadas *character blocks*). Estas teselas se referencian por un número y se disponen en el mapa, indicando qué tesela se dibuja en cada coordenada del tablero. Y lo mejor, que con solo cambiar el índice de la tesela a dibujar en cada casilla, será el *hardware* de la NDS el que actualizará el contenido de la pantalla.

Cada pantalla de la NDS puede mostrar cuatro capas (*backgrounds* o *BG*) que se nombran como BG0 a BG3 (para una pantalla) y SUB_BG0 hasta SUB_BG3 (para la otra). En la Figura 4[2] podemos ver una posible asignación de la correspondencia entre las pantallas y la memoria de vídeo de la NDS (VRAM), que es la que desarrollaremos a continuación. Para dibujar en las pantallas de la NDS habrá que llevar las teselas a la VRAM y asignar el mapa de correspondencias de las teselas en VRAM con el tablero de 32x24 casillas de cada pantalla y el BG correspondiente.

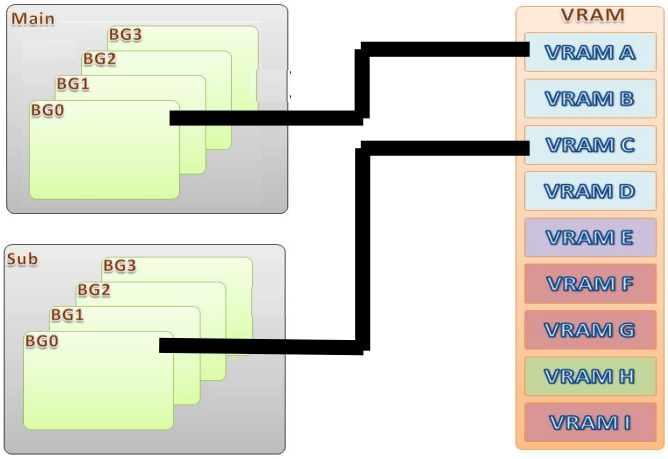


Figura 4: Asignación de capas o fondos a bancos de memoria de la VRAM. Fuente: véase nota 2 a pie de página .

4 Ejemplo práctico

El ejemplo que estamos reconstruyendo va a mostrar un mensaje de texto en una pantalla de la NDS, con un tipo de letra que vamos a diseñar, y un escenario de un bosque con un castillo en el centro. Para ello, necesitamos una imagen con el conjunto de caracteres que vamos a utilizar, y otra con los elementos del escenario.

Para empezar vamos a crear la imagen que contendrá el mapa de caracteres para los mensajes de texto. Se utilizará un tipo de letra que especificamos nosotros. Este mapa de caracteres lo podemos crear, p. ej., con el GIMP. En él podemos generar una imagen de 760 píxeles de ancho por 8 píxeles de alto, véase Figura 5a.

Con el menú *Imagen* | *Colorear...* accedemos a la ventana de diálogo para escoger un color personalizado, Figura 5b, donde seleccionaremos

[2] La imagen de la Figura 4 está basada en la de <http://4.bp.blogspot.com/_Vtd_Vdw1epI/R8f9k_A7R6I/AAAAAAAAAEs/piMMqJN18zs/s1600-h/nds_dg1.jpg >.

el color de valor hexadecimal “FF00FF”. Este, y podría haber sido cualquier otro, lo usaremos como color de transparencia para el fondo de los caracteres, Figura 5c.

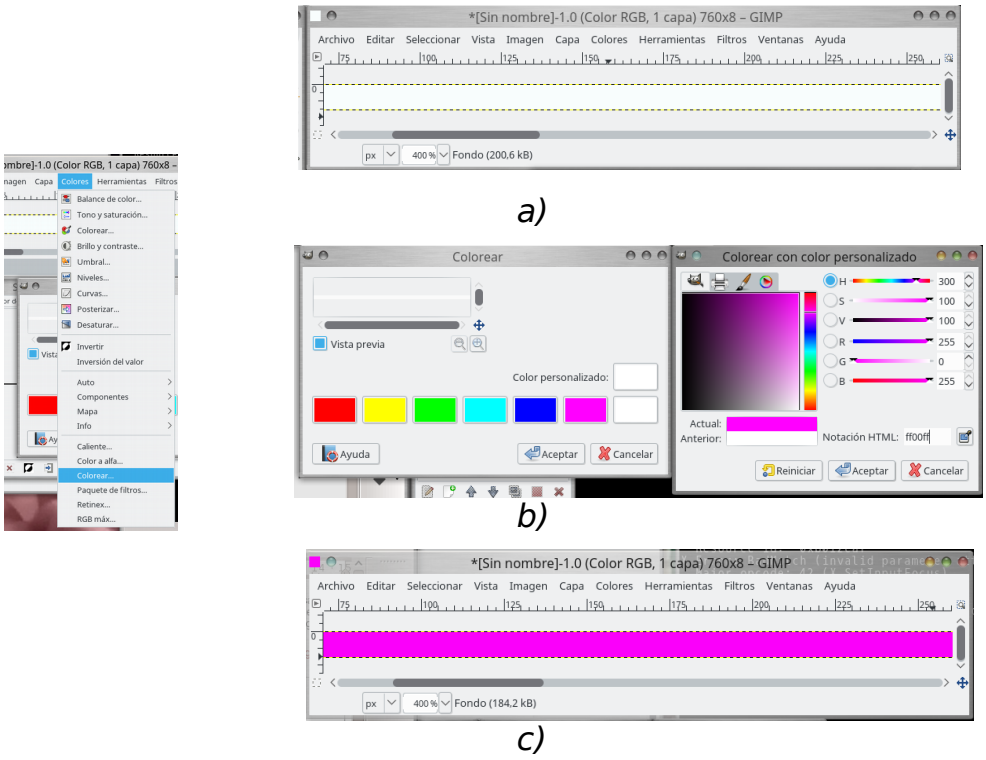


Figura 5: Cambiando el color de fondo de la imagen.

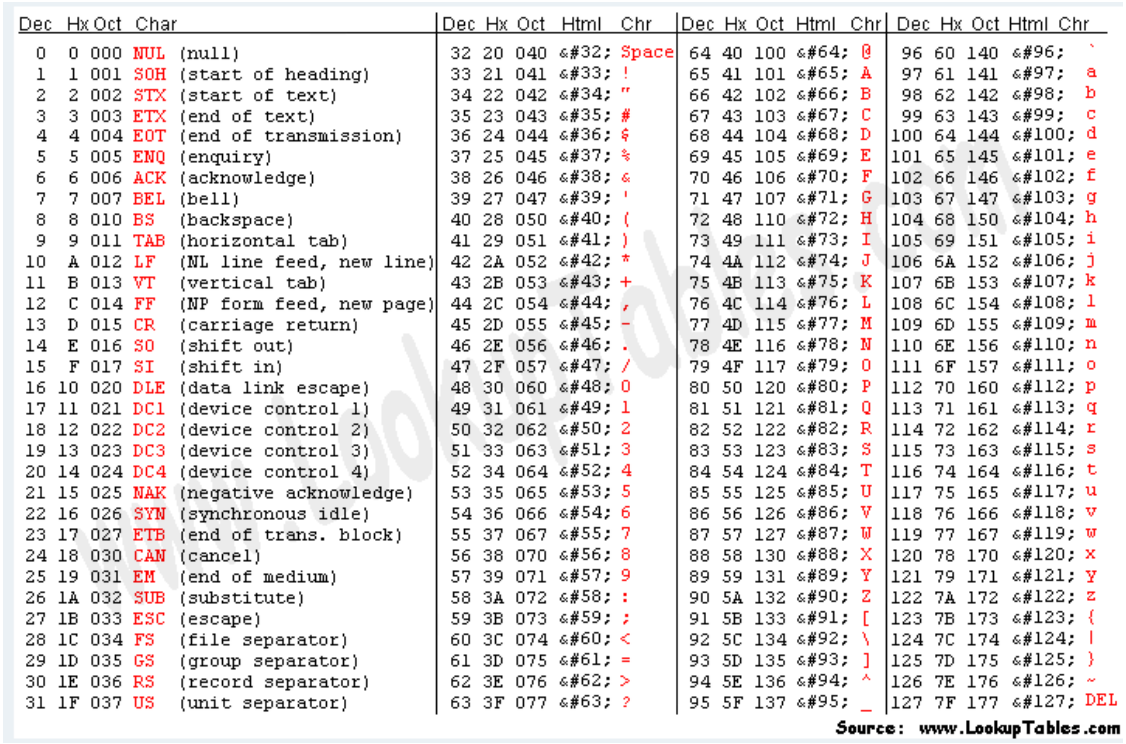
Que la imagen sea de 760 píxeles y que las teselas sean de 8x8, da lugar a que quepan 95 caracteres, lo cual nos da pie a incluir todos los caracteres imprimibles en pantalla (desde el espacio en blanco, de ordinal 32 en la tabla) del código ASCII original[3] básico, véase la Figura 6a. Aunque, si queremos utilizar más caracteres, como la ‘ç’ o la ‘ñ’, que están en la tabla ASCII extendida, Figura 6b, deberíamos construir una imagen más ancha. Lo dejamos a la curiosidad del lector interesado.

A lo que íbamos, ¿cómo generamos la imagen de la fuente de letra? Primero ampliamos la imagen, porque tendremos una muy fina y alargada. Vamos al menú *Vista* | *Ampliación* > *16:1* y le pediremos que nos muestre la rejilla para ver cómo los caracteres ocupan las teselas con *Imagen* | *Configurar la rejilla* … donde fijaremos, en “Espaciado”, la *Anchura* y *Altura* a 8 píxeles (Figura 7a), y el color de la misma. Ahora ya podemos ir al menú *Vista* | *Mostrar rejilla,* y tendremos sobrepuesta la rejilla a nuestra imagen (Figura 7b). Esta nos servirá para valorar si un tipo de letra nos sirve para esta labor, en tanto en cuanto, al escribir el alfabeto, veamos que los símbolos encajan correctamente en las casillas que muestra la rejilla.

Ahora, seleccionaremos el modo de inserción de texto con el menú *Herramientas* | *Texto,* y empezaremos por la esquina superior izquierda, Figura 8a. En la pequeña caja de texto que aparece dentro de la imagen, tendremos que cambiar la fuente de letra que pueda aparecer por una de tamaño 8 píxeles, como se muestra en la Figura 8b. Los caracteres

[3] Se puede ver el contenido de esta tabla, p. ej., en la URL <http://www.asciitable.com/>.

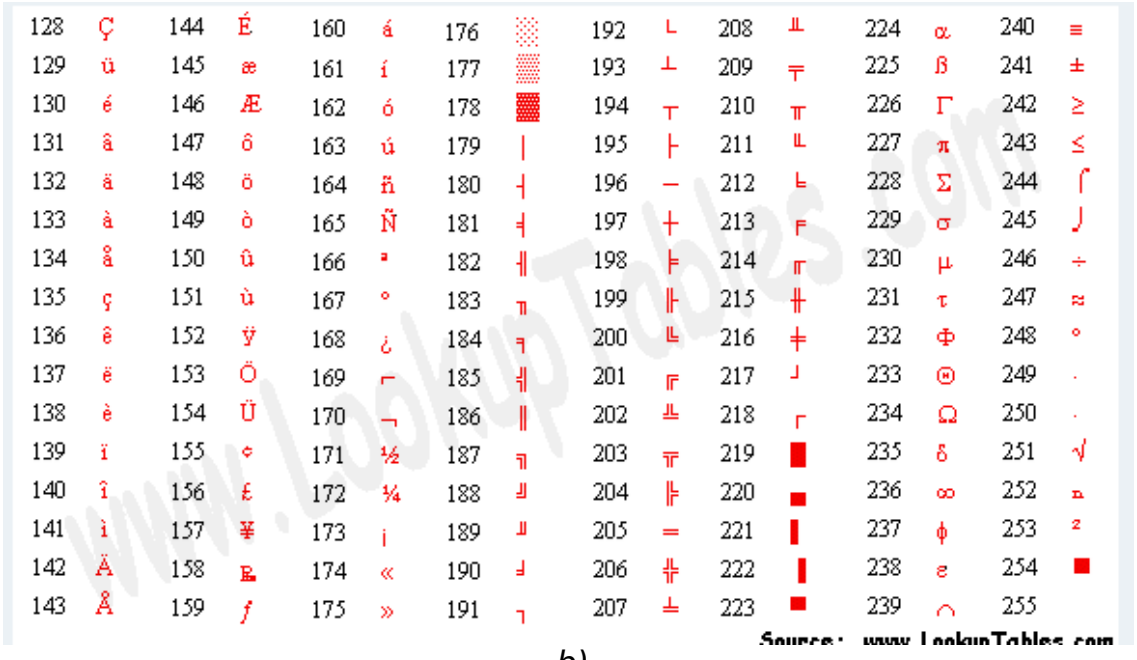
monoespaciados encajan mejor con una cuadrícula que asigna el mismo ancho a cada carácter, pero se puede utilizar cualquier tipo que podamos ajustar al alto de la imagen.



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

a)

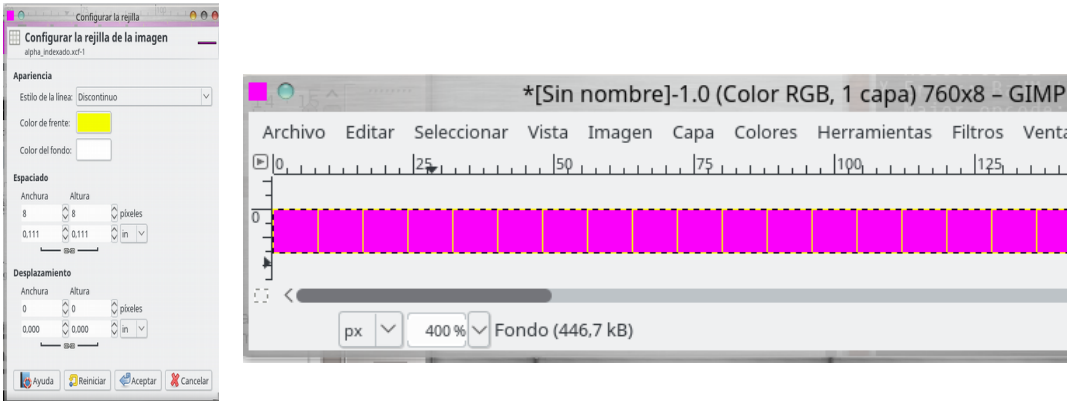


128	Ç	144	É	160	á	176	▒	192	└	208	╨	224	α	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
131	â	147	ô	163	ú	179	|	195	├	211	╙	227	π	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	·
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	█	235	δ	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	n
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	

Source: www.LookupTables.com

b)

Figura 6: Contenido de la tabla ASCII: (a) básica y (b) extendida. Fuente <http://www.asciitable.com/>



a) *b)*

Figura 7: Utilizar una rejilla para validar el teselado del mapa de caracteres: a) configurar la rejilla y (b) mostrarla.

a) *b)*

Figura 8: Generar el mapa de caracteres: (a) cuadros de diálogo que aparecen al seleccionar la herramienta de texto y (b) dos posibles tipos de letra.

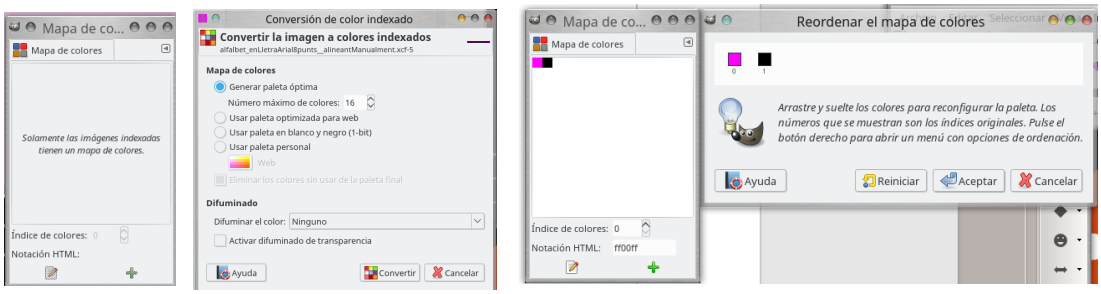
También es el momento de escoger el color de los caracteres, lo hemos cambiado de blanco a negro (pero, también podríamos haber escogido otro). Ahora, ya podemos escribir los caracteres siguiendo el orden de la tabla ASCII, empezando por el espacio en blanco (ordinal 32 de la tabla, veremos utilizado este valor en el código que hace posible que se utilice este tipo de letra en pantalla) y vamos escribiendo la tira de caracteres que mostramos entre comillas:

“ !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~”

Hemos hecho la letra más pequeña para que quede claro que debe escribirse todo en una línea, ¡sin las comillas del principio y del final!

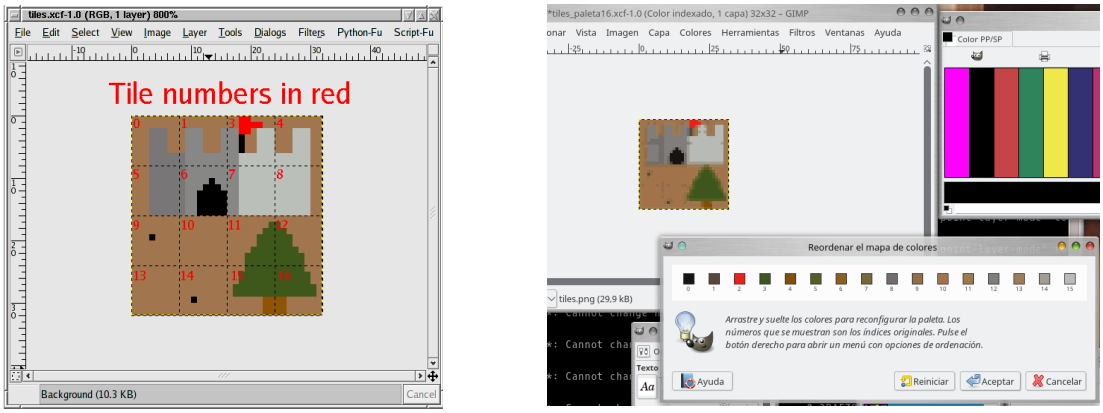
Una cosa más, hemos de generar una paleta para nuestra imagen de texto en la que los dos primeros colores sean el de fondo y el de color de los caracteres. Para verla, utilizaremos la entrada de menú *Ventanas* | *Diálogos empotrables* > *Mapa de colores*, Figura 9a. Tranquilos, si no la tiene es porque GIMP, en principio, crea una imagen sin paleta. Para obtenerla utilizaremos el menú *Imagen* | *Modo indexado ...* y, utilizando la opción “Generar paleta óptima” con 16 colores, Figura 9b, podremos

pulsar en el botón “Convertir” (Figura 9c) y obtenerla. Incluso, podríamos editar la paleta ahí mismo. Esta imagen ya puede ser exportada, p. ej. en formato PNG, para ser incluida en nuestro proyecto.



a) b) c)

Figura 9: Paleta de color: (a) Modo RGB, (b) generando la versión con paleta (indexada) de la imagen y (c) visualizando la paleta.



a) b)

Figura 10: Mapa de bits para el escenario del juego: (a) Indicando la numeración de las teselas (fuente [1]) y (b) generando la paleta de color.

Respecto a la imagen de fondo de la escena donde se desarrolla el juego, destacaremos que se puede ver descompuesta por teselas de 8x8 píxeles, que aparecen numeradas en la Figura 10a. Al contener 4x4 teselas, podemos calcular que el tamaño adecuado de este motivo debe ser de 32x32 píxeles y le aplicaremos también la conversión, Figura 10b, a imagen con paleta (indexada) de 16 colores. Con todos estos elementos ya podemos empezar a pensar en construir un diseño como el que muestra la Figura 11. Ya tenemos el escenario, pero el lector puede introducir su propia creatividad y hacer las variaciones que considere sobre él.

5 Conclusiones y cierre

Con este artículo hemos actualizado la secuencia de pasos para generar los materiales gráficos con que construir un escenario a base de teselas, para utilizarlo en el desarrollo de aplicaciones sobre la plataforma NDS. Las actualizaciones necesarias del código y la ubicación en un repositorio en *GitHub* [2] son nuestra modesta aportación.

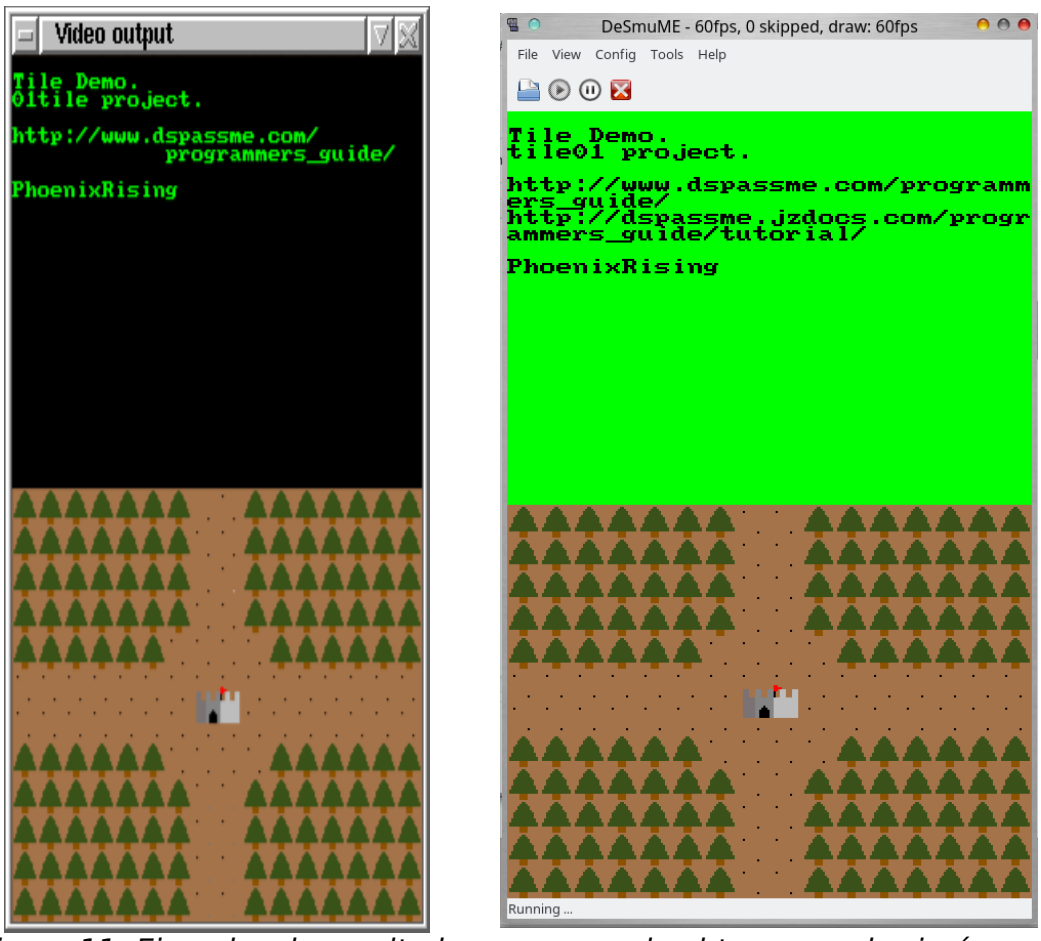


Figura 11: Ejemplos de resultado que se puede obtener con las imágenes aquí construidas. Imagen de la izquierda extraída de [1].

Queremos agradecer desde aquí el trabajo del autor original del artículo. *P. Rising*. Sirva este trabajo como homenaje a este y otros desarrolladores que han contribuido con su código, y con sus explicaciones, a que otros puedan adentrarse en este apasionante y difícil campo de desarrollo de aplicaciones para computadores, con una arquitectura tan especializada y poco documentada.

Esperamos que el lector se anime a descargar el proyecto desde el *GitHub* y, por supuesto, experimentar y modificar el código que se publica en el repositorio del proyecto.

6 Bibliografía

[1] P. Rising. (2005). *Homebrew Programmers Guide to the Nintendo DS* . Disponible en <http://dspassme.jzdocs.com/programmers_guide/tutorial/index.html >.

[2] Ejemplos de desarrollo para NDS. Repositorio en GitHub. Disponible en <https://github.com/magusti/NDS-hombrew-development >.

[3] DeSmuME. Página web del proyecto. Disponible en <http://desmume.org/>.

[4] J. Amero (Patater). (2008). *Introduction to Nintendo DS Programming*. Disponible en <https://patater.com/files/projects/manual/manual.html >.

[5] O. Boudeville. (2008). *A guide to homebrew development for the Nintendo DS*. Disponible en <http://osdl.sourceforge.net/main/documentation/misc/nintendo-DS/homebrew-guide/HomebrewForDS.html>.

[6] F. Moya y M. J. Santofimia. (2011). Laboratorio de Estructura de Computadores empleando videoconsolas Nintendo DS. Ed. *Bubok Publishing*. ISBN. 978-84-9981-039-3.