Witched

BE OR BE DEAD

Un juego de Amstrad CPC para 2 jugadores

MAKING OF

Código y gráficos | Hinaman
Música | Pioj
Traducciones | Susana Tapia
Pantalla de carga | MHR

Octubre 2022
#bewitchedobd

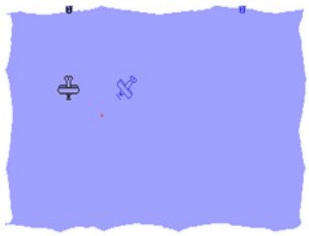
INTRODUCCIÓN

Mis mejores momentos con el Amstrad siempre fueron los que compartía jugando con amigos o primos con títulos como Ikari Warriors, Gauntlet o Golden Axe a los que podíamos jugar simultáneamente. Todas las risas (y algún que otro enfado) venían cuando se trataba de una competición entre los 2 jugadores, en muchos de esos recuerdos aparecen títulos como Super Dogfight de Commodore64, en el que perseguías el avión del rival para derribarlo. De ahí nace la idea de crear un juego para 2 jugadores con el que se pudieran revivir todos esos momentos.

Ikari Warriors (1986) Elite Software

Gauntlet (1985) U.S. Gold

Golden Axe (1990) Virgin Games, Probe Software


Super Dogfight (1983) R.V. Stevens

REQUISITOS

La idea principal del juego es que debía cumplir los siguientes requisitos:

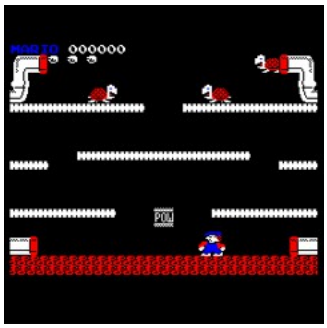
- 2 jugadores simultáneos
- Competición entre ambos, al estilo 'seek and destroy'
- Pantalla fija con plataformas para poder moverse por el escenario
- Algún tipo de arma o disparo para lanzar al otro jugador
- Posibilidad de 1 jugador contra la máquina (finalmente descartado)

REFERENTES

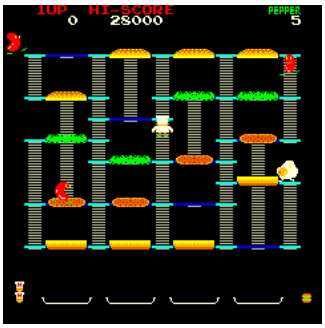
Teniendo en cuenta los requisitos anteriores se han utilizado como referencia algunos juegos que cumplen todos o por lo menos alguno de ellos:

- Super Smash Bros Brawl
- Mario Bros
- Burger Time

Super Smash Bros Brawl (2008) Game Arts, Sora Ltd., HAL Laboratory



Mario Bros (1987) Choice Software



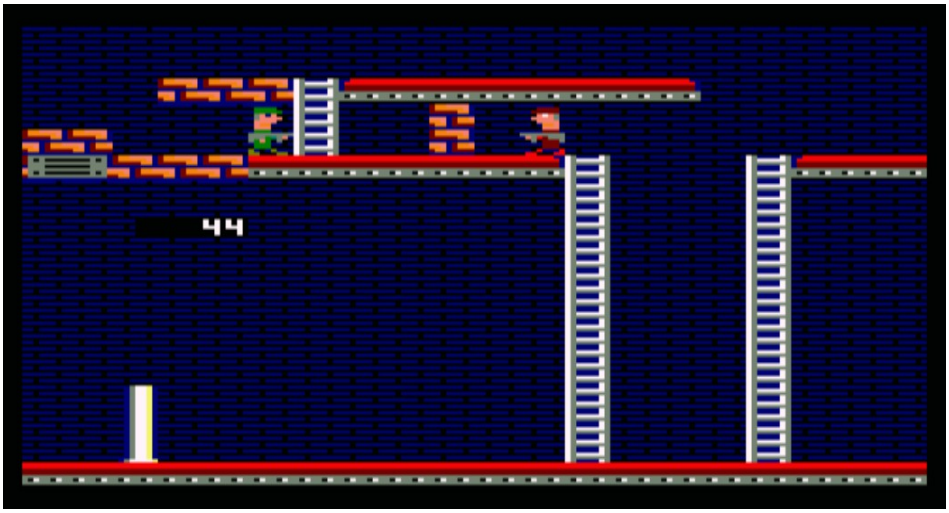
Burger Time (1982) Data East

DESARROLLO

Tras un par de intentos de iniciarme en el desarrollo de juegos para el Amstrad mediante CPCtelera y dejarlo de lado en ambas, me decido a tomarlo más en serio gracias a los videos del Profesor Retroman y a la serie de videos del Taller CPCtelera de Carlos Pérezgrín. Así el proyecto comienza a tomar forma en el verano de 2022.

Un primer boceto del juego tenía una ambientación bélica de disparos entre los jugadores.

Primer boceto de los sprites del protagonista



Primer prototipo "jugable"

PANTALLA Y GRÁFICOS

Se define la siguiente paleta de colores, 16 colores en modo 0:

Paleta

PALETA															
pen	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
color																
FW_COLOR	0	1	3	6	9	10	11	12	13	15	16	18	21	24	25	26

Tamaño del escenario

	Ancho					Alto			
	Tiles	Píxeles	Bytes	Píxel inicial	Píxel final	Tiles	Píxeles	Píxel inicial	Píxel final
PANTALLA		160	80	0	159		200	0	199
Marcador superior		160	80	0	159		48	0	47
Zona de juego	20	160	80	0	159	18	142	48	190
Zona inferior (mensajes)		160	80	0	159		9	191	199

Coordenadas para los sprites

COORDENADAS	MIN	MAX	PANTALLA
Jugador.X	AREA_JUEGO_IZQ 0	76	AREA_JUEGO_DER 80
Jugador.Y	AREA_JUEGO_SUP 48	174	AREA_JUEGO_INF 190

Tamaño de sprites y tiles

	Ancho		Alto
Sprite	Píxeles	Bytes	Píxeles
Jugador	8	4	8
Tile	8	4	8
Hechizo	8	4	7
Item	8	4	8

ESCENARIOS Y TILES

Los escenarios están formados por 20 tiles de ancho y 18 de alto, creados con Tiled. Los mapas se guardan comprimidos y cuando se van a utilizar se descomprimen en un único array (mapa[20*18]) del que ya se ha reservado la memoria.

Por el espacio disponible se han conseguido añadir 5 escenarios diferentes, y en el menú de selección de mapa deja elegir un sexto mapa que hará que se seleccione un mapa aleatorio de entre esos 5 cada vez que comience la partida.

Los tiles que forman el mapa son de tamaño 8x8 píxeles, y para poder diferenciarlos se ordenan de esta forma:

1- Tiles de fondo (no interactúan con los jugadores)
2- Tiles especiales (pinchos y activador de trampas)
3- Escaleras (permiten subir o bajar y no dejan caer al jugador)
4- Plataformas (no dejan caer al jugador pero no son obstáculo horizontal ni destruyen los hechizos lanzados)
5- Obstáculos (detienen al jugador y destruyen los hechizos lanzados al colisionar)

Sprite de los 27 tiles disponibles en el juego

En los primeros bocetos de los tiles incluía el guiño al juego
Lárcenas con retratos del *constructor* y el *banquero* que aparecían como fondo en los escenarios

Las coordenadas de inicio de los jugadores y las de los items de cada escenario se han definido en en archivo 'objetos.h', donde por cada nivel establecemos las coordenadas X e Y de los 2 jugadores y las posibles coordenadas donde pueden aparecer los items.

Al comenzar la partida se leen los registros correspondientes al mapa actual. Para programar esta forma de leer los datos de mapa me he basado en los tutoriales de Carlos Pérezgrín y su juego Anima.

SPRITES Y PERSONAJES DEL JUEGO

Durante el juego podrás controlar los siguientes personajes:

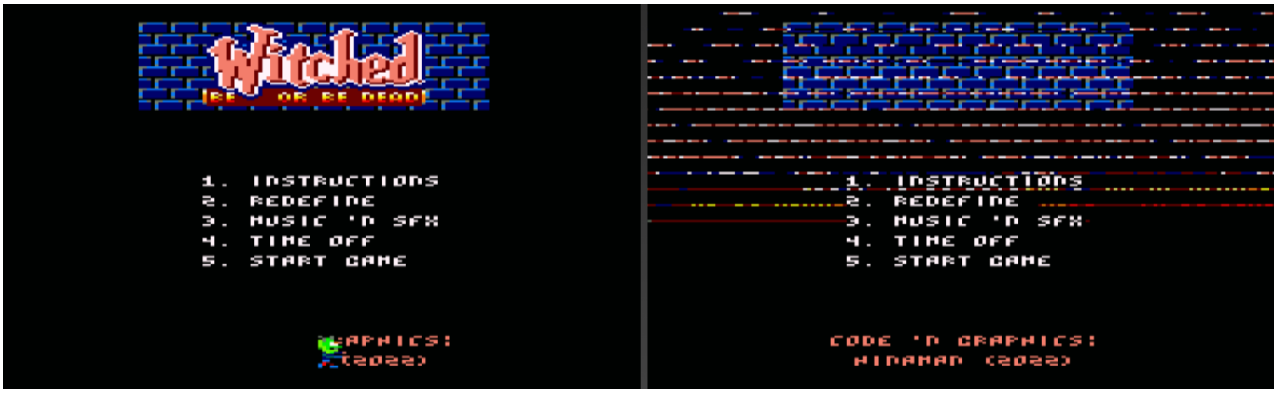
	ED SPELLS 

	WITCHER 
	ZOMBIE 
	FROGGY 
	MAGIK MAX 
	LÁRCENAS 

Para animar los personajes se intercalan los fotogramas y se utilizan las funciones de CPCTelera para voltearlos y hacer que miren a izquierda o derecha.

Los sprites y las caras de los personajes se almacenan comprimidos para optimizar el espacio, y se descomprimen durante el juego cuando son necesarios en una zona de memoria ya reservada.

Con el logo del juego que aparece tanto en el menú como en la pantalla de demostración de escenarios se utiliza una técnica similar. Ya que en los menús sólo se utiliza un buffer de video, el gráfico se descomprime en el segundo buffer (invisible en ese momento) y se copia al primer buffer (el visible) para poder mostrarlo sin producir un coste adicional de memoria.



A la izquierda se muestra el buffer de video visible en los menús, y a la derecha el que no se ve.
Se aprecia como el logo se ha descomprimido en el derecho y se muestra en el izquierdo.
Igualmente el sprite del personaje que aparece borrando los créditos se muestra únicamente en el primer buffer de video.
Los textos y el fondo de ladrillos se muestran en ambos buffers por optimización de código, pero podrían mostrarse solamente en el primero.
(Imágenes obtenidas gracias a las herramientas de Retro Virtual Machine)

FUENTES PERSONALIZADAS

Para escribir los textos del juego he utilizado un sprite con todas las fuentes necesarias (unos pocos símbolos, números y letras mayúsculas), de tamaño 4x5 píxeles cada carácter, y colocadas verticalmente una por debajo de otra para que sea más fácil 'recortarlas' cuando se escriben textos en pantalla.

La función printTxt lee el texto que se quiere escribir y obtiene su valor decimal ASCII correspondiente para dibujar en pantalla el

sprite del carácter. Como se utiliza doble buffer durante el juego y para ahorrar código, la función escribe siempre el texto en ambos buffers de video.

Además la función cuenta con un parámetro mediante el que le podemos indicar de qué color queremos la fuente, y mediante las funciones de CPCTelera modificamos el color blanco original por el indicado.

El tamaño de la fuente es bastante pequeño, lo que nos hace ahorrar un poco de memoria, pero se puede leer sin muchos problemas.

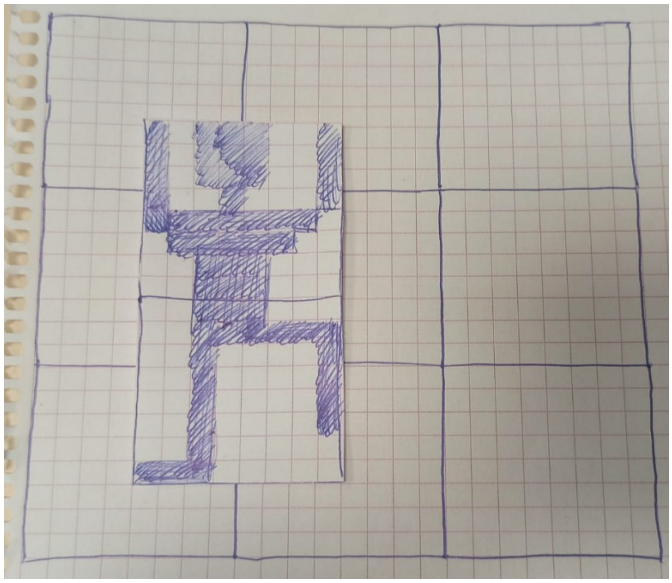
Ejemplo de texto

MOVIMIENTO DE LOS PERSONAJES

El sistema de movimiento de los personajes y colisiones con las plataformas/tiles es la base de este tipo de juegos, según la posición del sprite se debe tener en cuenta:

- Si el personaje está sobre un obstáculo o plataforma que evite que caiga.
- Si el personaje está en contacto con una escalera hará que se pueda subir o bajar.
- Para crear las animaciones se necesita redibujar el fondo, teniendo en cuenta el número de tiles que deberán redibujarse.

Ya que el tamaño de los tiles que forman el mapa son de 8x8 píxeles y que los personajes tienen un tamaño de 8x16 píxeles, lo más práctico es crear una plantilla en papel para poder ver fácilmente cuántos tiles de ancho y de alto ocupa el sprite del jugador dependiendo de su posición en el escenario:



Moviendo el personaje sobre el papel podemos ver cuántos tiles ocupa y programar su redibujado en cada caso.

COLISIONES CON EL ESCENARIO

Para revisar las colisiones de los protagonistas con el escenario seguí este esquema para averiguar el tipo de tile del mapa con el que se va a colisionar (muro, escalera, fondo...), y así saber si le permitimos moverse o no, subir escalera o no o si debe caer.

Según la posición del sprite con los tiles del escenario se consulta una tabla u otra para averiguar el tipo de tile de cada coordenada:

- PosX y PosY hacen referencia a la coordenada del vértice superior izquierda del sprite
- SPRITE_W y SPRITE_H hacen referencia al tamaño del sprite
- TILE_W y TILE_H hacen referencia al tamaño de los tiles

PosX – SÍ Coincide con el inicio de un tile
PosY – SÍ Coincide con el inicio de un tile

¿Qué queremos conocer?	Zona del sprite	¿Cuál será la coordenada X a consultar?	¿Cuál será la coordenada Y?
Tipo de tile a la IZQ	1/2 superior	PosX - TILE_W	PosY
	1/2 inferior	PosX - TILE_W	PosY + TILE_H
Tipo de tile a la DER	1/2 superior	PosX + SPRITE_W	PosY
	1/2 inferior	PosX + SPRITE_W	PosY + TILE_H
Qué se pisa	Debajo	PosX	PosY + SPRITE_H
Si estamos en una escalera	1/2 inferior	PosX	PosY + TILE_H

PosX – SÍ Coincide con el inicio de un tile
PosY – NO Coincide con el inicio de un tile

¿Qué queremos conocer?	Zona del sprite	¿Cuál será la coordenada X a consultar?	¿Cuál será la coordenada Y?
Tipo de tile a la IZQ	1/3 superior	PosX - TILE_W	PosY
	Parte central	PosX - TILE_W	PosY + TILE_H
	1/3 inferior	PosX - TILE_W	PosY + (2 * TILE_H)
Tipo de tile a la DER	1/3 superior	PosX + TILE_W	PosY
	Parte central	PosX + TILE_W	PosY + TILE_H
	1/3 inferior	PosX + TILE_W	PosY + (2 * TILE_H)
Si estamos en una escalera	Parte central	PosX	PosY + TILE_H
	1/3 inferior	PosX	PosY + (2 * TILE_H)

PosX – NO Coincide con el inicio de un tile
PosY – SÍ Coincide con el inicio de un tile

¿Qué queremos conocer?	Zona del sprite	¿Cuál será la coordenada X a consultar?	¿Cuál será la coordenada Y?
Qué se pisa	Debajo (tile izquierda)	PosX	PosY + SPRITE_H
	Debajo (tile derecha)	PosX + TILE_W	PosY + SPRITE_H
Si estamos en una escalera	1/2 inferior (tile izquierda)	PosX	PosY + TILE_H
	1/2 inferior (tile derecha)	PosX + TILE_W	PosY + TILE_H

PosX – NO Coincide con el inicio de un tile
PosY – NO Coincide con el inicio de un tile

¿Qué queremos conocer?	Zona del sprite	¿Cuál será la coordenada X a consultar?	¿Cuál será la coordenada Y?
Si estamos en una escalera	Solapado (tile izquierda medio)	PosX	PosY + TILE_H
	Solapado (tile izquierda inferior)	PosX	PosY + SPRITE_H
	Solapado (tile derecha medio)	PosX + TILE_W	PosY + TILE_H
	Solapado (tile derecha inferior)	PosX + TILE_W	PosY + SPRITE_H

Nota mental: Para el próximo juego evitar las escaleras...

USO DE LA MEMORIA DEL CPC

Para evitar el parpadeo en el redibujado de los sprites se ha utilizado un doble buffer de video. El inconveniente es que dedicamos el doble de espacio de memoria para el video, lo que nos reduce el espacio de memoria que podemos dedicar al código del juego.

También se debe tener en cuenta reservar un espacio de memoria para la tabla de transparencias que necesita CPCTelera para dibujar los sprites transparentes. Además de las posiciones de las canciones de Arkos Tracker, dónde situar el inicio del código del programa y dónde debe situarse el stackLocation.

En el archivo memory.c defino algunos sprites para colocarlos manualmente en posiciones concretas de la memoria y así rellenar huecos para aprovecharla mejor.

En el siguiente esquema muestro la división de la memoria que utiliza el juego, esquema muy útil para tenerlo durante el desarrollo y reubicar los espacios reservados:

Direcciones de memoria inicio y fin (hex) Tamaño (bytes en decimal)	Uso
0xFFFF 0xC000	Primer buffer de video VMEM_BUFFER1
0xBFFF 0x8000	Segundo buffer de video VMEM_BUFFER2
0x7FFF 0x0BDB	↓ StackLocation ↑ Código
0x0BDA Size: 728 0x0903	Música: m_bewitch
0x0902 Size: 951 0x054C	Música: m_ingame
0x054B Size: 339 0x03F9	Música (efectos de sonido): m_sfx
0x03F8 Size: 249 0x0300	Música: m_gameover
0x02FF Size: 255 0x0200	Tabla de transparencia
0x01FE Size: 28 0x01E2	Sprite: sp_hechizo_1
0x01E1 Size: 28 0x01C5	Sprite: sp_hechizo_0
0x01C4 Size: 28 0x01A8	Sprite: sp_corazon
0x01A7 Size: 32 0x0187	Sprite: sp_arrow_small_left
0x0186 Size: 64 0x0146	Sprite: sp_efecto_transforma
0x0145 0x0040	Loader para el CDT

MÚSICA

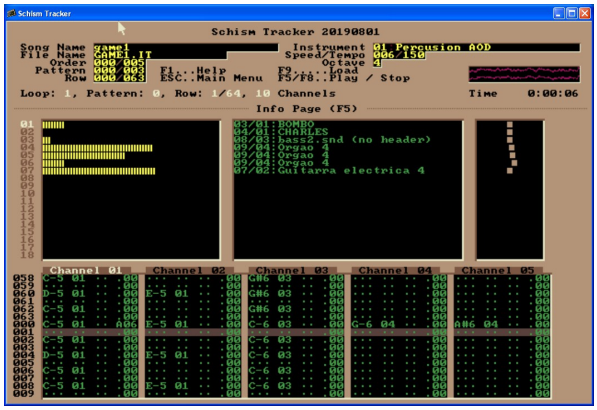
Para las músicas conté con la colaboración de Pioj, militante de C64, Unity, gran amigo, compañero de experiencias musicales y con bastante mejor oído musical que el mío.

El objetivo era tener 2 melodías, una para el juego y otra para los menús (siempre que tuviera suficiente memoria libre en el CPC). Unas melodías de unos 20 o 30 segundos que se pudieran repetir en bucle.

Para el juego quería alguna melodía machacona del estilo Solomon's Key (casi nada...). Aunque la música finalmente no tiene mucho que ver, sí que ha resultado una melodía muy pegadiza que encaja muy bien con el juego.

Para la música de los menús tenía muy clara mi petición. Llamándose el juego "Be witched o be dead" la melodía debía ser cañera con regusto a Iron Maiden, y esa fue nuestra fuente de inspiración.

Las canciones se han creado con Arkos Tracker 1 para poder utilizarlas con CPCTelera, si bien, el trabajar directamente con Arkos Tracker puede resultar muy frustrante con ordenadores actuales debido a los continuos cuelgues de la aplicación. Para evitar ésto las canciones se crearon con Schism Tracker para después traspasarlas a mano (nota a nota) a Arkos Tracker, con menos riesgo de perder el trabajo ya hecho.



Schism Tracker (clon de Impulse Tracker)
Con la documentación se incluye un pequeño archivo .ogg con un primer boceto del tema del juego compuesto con Schism Tracker

Como consejo, si quieres editar con Arkos Tracker 1 sobre Windows: instala una máquina virtual con Windows XP y trabaja sobre ella.

IDIOMAS Y TRADUCCIONES

Otra de las ideas originales era crear el juego en 2 idiomas (castellano e inglés) en una misma carga, pero la falta de tiempo (y planificación desde el inicio con el código) y la poca memoria del CPC me llevaron a utilizar únicamente el inglés. Las pocas frases que aparecen en el juego no suponen un impedimento para que cualquiera pueda jugarlo aunque no conozca el idioma, aunque sí que está disponible el manual en los 2 idiomas en archivo PDF.

Para asegurarme que no cometía muchos errores con el idioma conté con la ayuda de Susana Tapia, y no podría estar más agradecido y satisfecho de su trabajo.

PANTALLA DE CARGA

Si había algo que enganchaba a la pantalla del Amstrad era ver cómo se iba dibujando línea a línea la imagen de carga, rodeada de unos bordes de colores hipnóticos. Para el juego quería mantener esto aunque hiciera que el juego tardara un poco más en cargar (al no utilizar una imagen comprimida como permite hacer CPCTelera). Para eso se ha creado un segundo proyecto (en la capeta Loader) que se encarga de generar la carga para la cinta en formato CDT.

La imagen de la pantalla de carga es cortesía de MHR, apasionado de la fotografía y de la edición digital. Él editó la imagen en tiempo récord, un crack, que más puedo decir... te quiero papá.

Imagen de la pantalla de carga.
Para convertir y ajustar la imagen al Amstrad se ha utilizado ConvImgCPC

GUIÑO CPCRETRODEV 2022

Para cumplir con las bases de la #CPCRetroDev2022 el juego debe incluir un guiño al ganador de la primera edición del concurso: Larcena's Legacy de Los Pollos Amigos. Por lo que una de las posibles transformaciones de los protagonistas será del banquero en *Lárcenas*, uno de los protagonistas, haciéndolo parte del *gameplay.*

Protagonistas del juego Larcena's Legacy

Aparición estelar del banquero en Be Witched Or Be Dead

CRÉDITOS

- Edu Hinarejos | Código y gráficos
- Pioj | Música y tests
- Susana Tapia | Traducciones
- MHR | Pantalla de carga

SOFTWARE UTILIZADO

- CPCTelera | Frameword de desarrolo para crear juegos y contenido para amstrad CPC
- Visual Studio Code | Editor de código
- Aseprite | Editor de sprites y pixel art
- Tiled | Editor de niveles
- Arkos Tracker | Editor musical de tipo tracker para música y efectos de sonido exportables al Amstrad CPC
- Schism Tracker | Editor musical de tipo tracker
- ConvImgCpc | Conversor de imágenes de PC a CPC
- playTXZ | Reproductor de archivos CDT para probar el juego en un Amstrad CPC real
- Retro Virtual Machine | Emulador de Amstrad CPC
- WinAPE | Emulador de Amstrad CPC
- OpenOffice | Editor de texto para la documentación
- Henny Penny de Brownfox | Fuente para "Witched" del título bajo Open Font License
- Imagen de la carátula del caset por jcoope12 bajo licencia Pixabay license

AGRADECIMIENTOS

- LADDH | Por las pruebas, ideas y su ilusión por lo retro
- José María Pardina | Testeos e ideas
- Maribel Fuentes | Atrezo para la pantalla de carga

REFLEXIÓN FINAL

Desarrollar este juego ha resultado un reto muy divertido. Ir descubriendo las herramientas, cómo funciona CPCTelera, la memoria del Amstrad y los nuevos conceptos como las interrupciones, todo ha hecho que la experiencia (en principio un poco frustrante) haya resultado muy gratificante.

El conocer la existencia de la CPCRetroDev y tener una fecha límite hace que te pongas las pilas. Ha sido un aprendizaje continuo, de ahí el posible caos en el código, el mezclar castellano e inglés en muchas partes, todo por no tener muy clara la idea global desde el inicio... todo un 'aprende mientras creas'.

Pero el poder presentar algo jugable y acabado es una enorme satisfacción, sobretodo cuando pasas una tarde con tu sobrina de 9 años enseñándole un Amstrad por primera vez y acaba diciéndote que tu juego le gusta más que el Arkanoid, sólo por eso ya ha valido la pena.

A todo color en el 464+

El juego funcionando en mi Amstrad CPC 464 de fósforo verde

Hinaman, Octubre de 2022