PENGUIN CROSS

MAKING OF

INTRODUCCION 2
¿COMO HA SIDO REALIZADO EL JUEGO? 2
TECNOLOGIAS UTILIZADAS 5
PROBLEMAS ENCONTRADOS 5
LECCIONES APRENDIDAS 5
EVOLUCION DE LOS SPRITES 6
- Jugador 6
- Meta 6
- Enemigos 6
- Plataformas 7
- Plataformas sumergidas 7
- Reloj 7
- Portales 7

INTRODUCCION

Penguin Cross es un videojuego para Amstrad CPC 464 programado en ensamblador por 3 estudiantes de Ingeniería Multimedia en la Universidad de Alicante.

Los autores del juego somos Francesc Bellido Delgado (paco.bellido.delgado@gmail.com), David Mulero Pérez (davidmuleroperez@gmail.com) y Wei He (weicocn1@gmail.com).

Somos **Cuboid Studio** y se puede contactar con nosotros vía Twitter @CuboidStudio

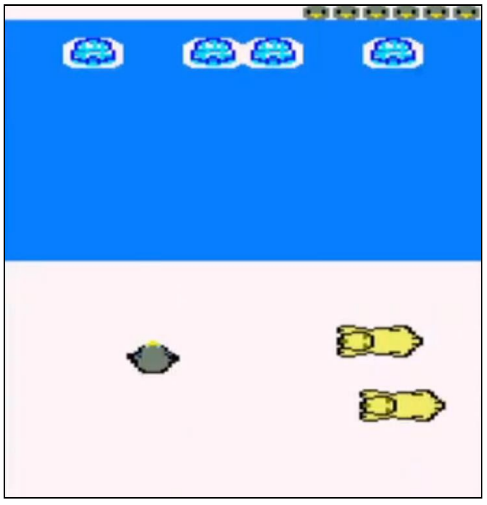
En este archivo vamos a explicar cómo ha sido programado durante las 5 semanas en las que ha sido desarrollado.

¿CÓMO HA SIDO REALIZADO EL JUEGO?

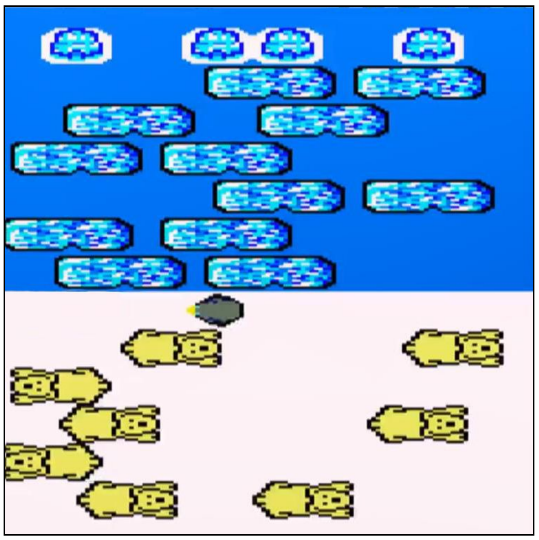
Para poder desarrollar el videojuego hemos hecho uso de lenguaje ensamblador. Durante 2 semanas empezamos a seguir cursos online para poder entender código máquina y empezar a hacer uso del lenguaje ensamblador. En primer lugar dibujamos algunos sprites a código máquina, programamos algunas animaciones en ensamblador y después comenzamos a hacer una estructura inicial ECS (Entidad-Componente-Sistema) para realizar un efecto Starfield.

Una vez que obtuvimos una buena base de aprendizaje comenzamos el desarrollo de nuestro juego que ha durado 5 semanas.

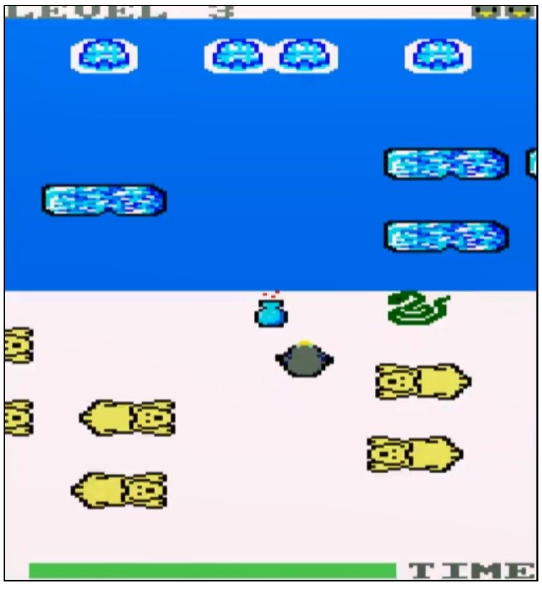
- En la primera semana realizamos un prototipo jugable donde un personaje principal podía moverse y colisionar con los enemigos. Además implementamos los iglús que funcionan como meta.



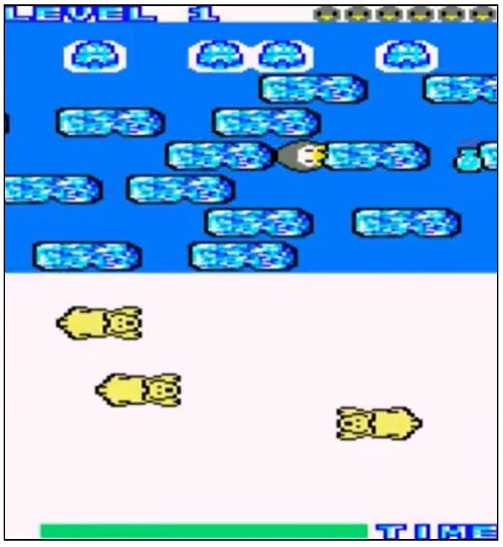
- En la segunda semana implementamos las plataformas a las que el jugador puede subirse para no morir ahogado en el agua y el sistema de generación de enemigos.



- En la tercera semana comenzamos el desarrollo de ítems, creamos el sistema de niveles, nuevos enemigos (mapache) y nuevo comportamiento para las plataformas e implementamos el clipping de los enemigos y el uso de la máscara para el dibujado de los ítems y del personaje principal.



- En la cuarta semana desarrollamos el doble buffer para evitar el flickering, implementamos las interrupciones para controlar el tiempo de juego e hicimos un primer contacto con la música.



- En la última semana hemos terminado la música, mejorado el game design, incluido enemigos y plataformas alternativas, implementado los menús (créditos, controles, game over) y redactado la documentación necesaria para poder presentar el juego a la CPCRetroDev 2020.

TECNOLOGIAS UTILIZADAS

- Utilizamos la librería **CPCtelera**, publicada bajo GNU GPL con Copyright (C) 2018 ronaldo / Fremos / Cheesetea / ByteRealms (@FranGallegoBR).
- Sublime Text 3 / Visual Studio Code
- Git Hub
- Arkos Tracker 1
- WinAPE/Retro Virtual Machine
- Gimp / Photoshop

PROBLEMAS ENCONTRADOS

Durante el desarrollo hemos encontrado algunas dificultades que han podido retrasar nuestro ritmo de programación. WinAPE nos ha causado algunos problemas como no poder abrirse o no aparecer las strings (en caso de abrir varias instancias del programa se corrompe la configuración y deja de haber ROM, la solución a este problema fue restablecer WinAPE). Además, era la primera vez que hacíamos uso de Git por lo que hemos tenido que aprender a usarlo paralelamente al desarrollo del juego.

En cuanto al desarrollo tuvimos varios problemas al reducir el tamaño de la pantalla ya que tuvimos que editar muchas de las funciones de CPCtelera, también en la implementación del doble buffer ya que tuvimos que cambiar muchas de nuestras funciones para poder hacer correctamente el cambio de buffer en cada momento. Para la implementación del clipping necesitamos realizar los cálculos para el borrado y nos llevó bastante tiempo realizarlo correctamente.

En algunas ocasiones nos hemos encontrado bugs que corrompían el juego (funciones que destruían registros, funciones que tuvimos que editar al realizar el doble buffer, la gestión de las interrupciones…) y tuvimos que realizar debug con WinAPE para poder solucionarlos.

LECCIONES APRENDIDAS

Durante el desarrollo de Penguin Cross hemos aprendido a entender mejor el lenguaje a bajo nivel. Durante las primeras semanas del curso comenzamos a estudiar código máquina y lenguaje ensamblador.

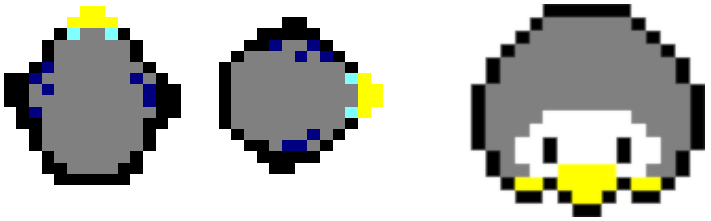
Además, hemos aprendido a hacer uso de *Git* y *GitHub*, lo cual nos ha sido muy útil para poder realizar el control de versiones de nuestro código y ha facilitado en gran parte el trabajo en equipo.

Por otro lado, hemos mejorado nuestra habilidad en la creación de sprites mediante Gimp/Photoshop y de música en 8 bits mediante Arkos Tracker 1.

EVOLUCION DE LOS SPRITES

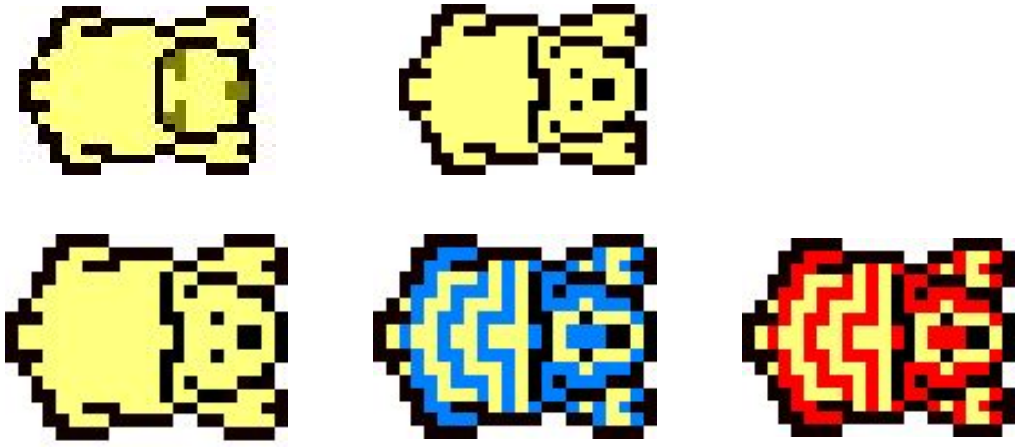
A continuación vamos a mostrar la evolución de algunos de nuestros sprites. La creación de sprites se ha mantenido durante todo el desarrollo y muchos de ellos han necesitado mejorar su versión inicial para poder lucir correctamente. Conforme mejoraba nuestra habilidad para crearlos ha sido más fácil crearlos con un mejor aspecto.

Jugador



Meta

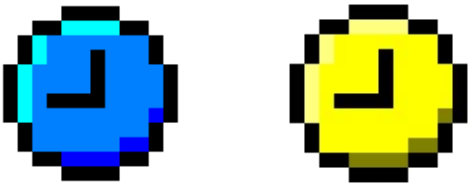
Enemigos



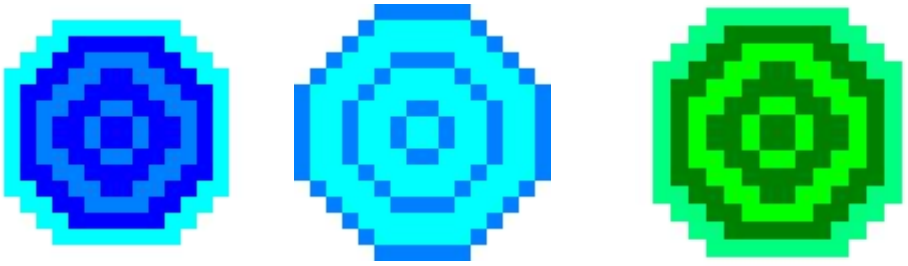
Plataformas

Plataformas sumergidas

Reloj



Portales



Fuente de texto