FARMING SPIRITS

TRUNCADO STUDIOS

MAKING OF

INDEX

About us

We are **Truncado Studios**, a game development studio created in September of 2022 as a studio used for game university projects. We are only 3 students.

- **Víctor Tébar Juan**
- **Lucía Oliva Arocas**
- **Luis Morata Gómez**

About “Farming Spirits”

We wanted to create a fun game. For this reason we created *Farming Spirits,* a survival game where you must collect the souls of spirits to become stronger.

Farming Spirits is developed for the **CPCRetroDev 2022.** The game is entirely developed in assembler code, using the Amstrad game engine **CPCtelera**. You can also run this game on a real **Amstrad CPC 464** machine.

Technologies used during the development of the game

- **Visual Code** → Text editor
- **CPCTelera** → Amstrad game engine
- **Arkos Tracker 1** → Musical program used for producing music
- **Aseprite** → Pixel art tool

Early stages of development

The idea

The idea was far from clear. In fact it took us a week and a half or so to have any idea what we were going to do.
But we knew it was going to be singleplayer.
At first we just used flat rectangles as sprites because the theme wasn't clear until a week before the game was delivered.
But the most important thing was to have a comfortable game engine to build the rest of the game on.

Middle stages of development

Render

To avoid any possible flickering or tearing we created a double buffer. In order not to run out of memory quickly we reduced the screen size to 32 characters wide x 16 characters high.
As there were going to be quite a few entities on screen at once, some things are only drawn once, like the background and the enemy platforms, and others only when they are modified, like the HUD.
For the images on the main screen, level change, etc. Obviously they were compressed so as not to waste all the memory on images.

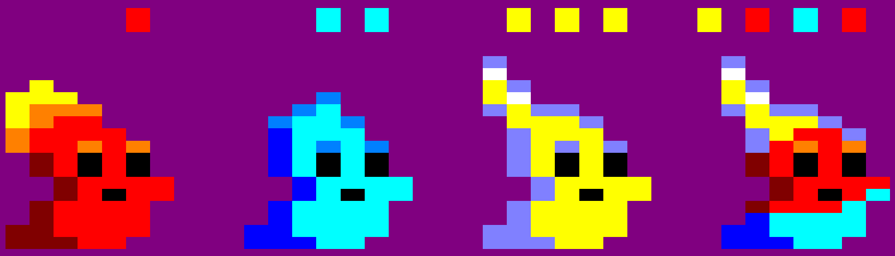
Late Stages of Development

Once we had a playable prototype it was time to start putting in the art to make it a friendly and beautiful experience.

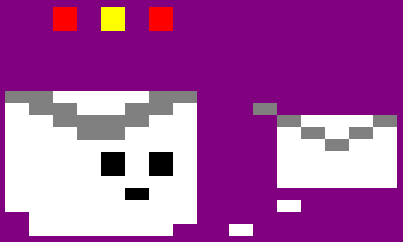
Art

Spirits

We created 3 basic spirits, 1 elemental and 2 mixed spirits. Almost all spirits are the same with some modifications due to space limitations and to keep consistency across all spirits.

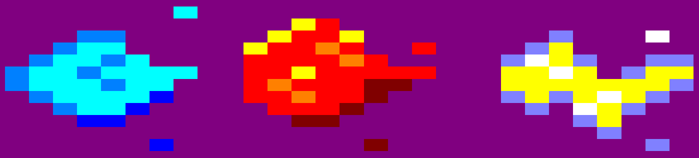


As a reference to Lacena's Lacery we made the letter.



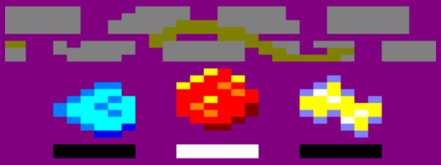
Spells

To clarify which spell is being used we used a monochromatic palette for each one.
In all of them animations were created to give the player as much feedback as possible.



Hud

Since we had so little space on the HUD, we had to think of a good way to represent the spells in a way that everyone could understand.



Wizard

As the main character we created a wizard.

Screens

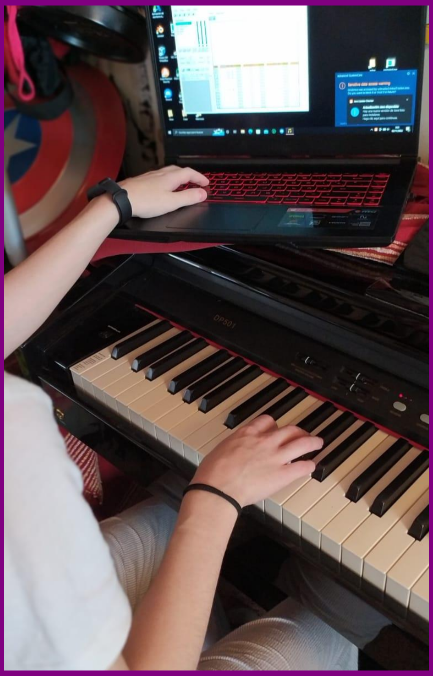
All screens of start, load, transition images are based on the same background screen for the game to maintain consistency. But we had to make them different enough for the player to know quickly where they were.

Music

For the music we knew that we wanted a short song that could be looped, and that would go through various moods. For this reason we chose to write it in D Dorian, because of its minor and epic nature.

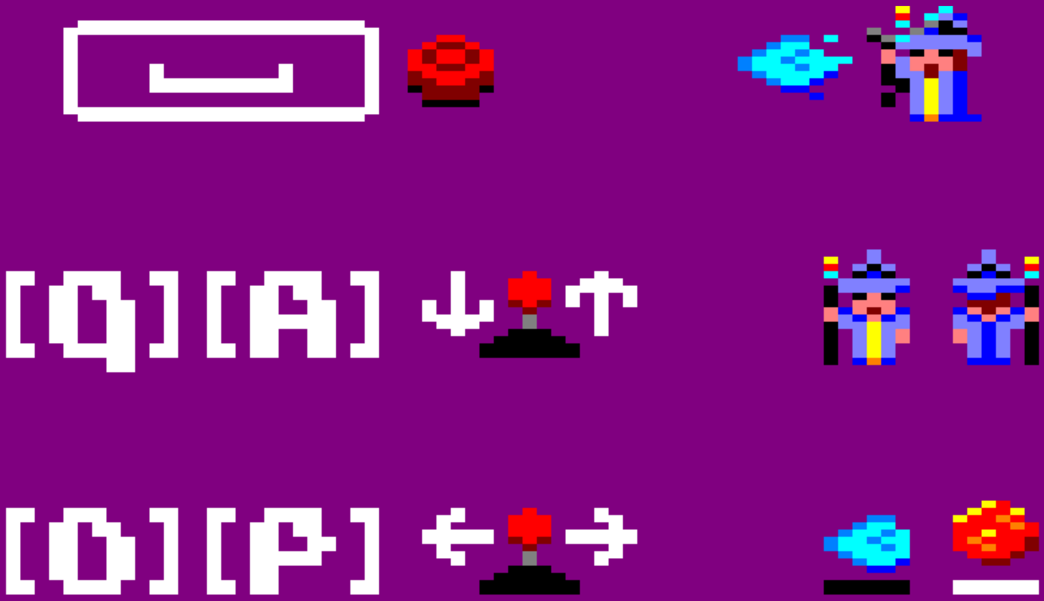
The menu needed more relaxed music that would match the game, so we decided to create a simple loop in A minor.

To compose it, we improvised on the piano and put directly into Arkos Tracker 1 whatever we liked. Using CPCtelera’s music utilities it was really easy to add music to our game.



User experience

We carried out several tests with people outside the project to check that the game could be understood without the need for explanations. It took several iterations of this process to make the game self-explanatory.
What we had the most problems with was the traffic lights that tell you when the line of enemies is about to advance.



Learned lessons

Low level programming is always a good option to know how programs really work. These are some of the most important lessons that we’ve learned:

- **Less is more**: doing lots of things, lots of mechanics is tempting, but with such a limited machine, so little experience and so little time, you have to concentrate on the things that are really important.
- **Debug data, not code**: what matters in your software is not the code, it’s the data. For debugging the fastest way by far is to see what is happening to the data instead of looking at your "perfect" code.
- **Communication**: any kind of team project without communication is intended to fail. It was a good tool knowing what everyone was doing. Knowing this, unexpected results didn’t happen. Letting your teammates know what you are doing and listen to them is the key to success.

We hope you enjoy Farming Spirits

- Truncado Studios