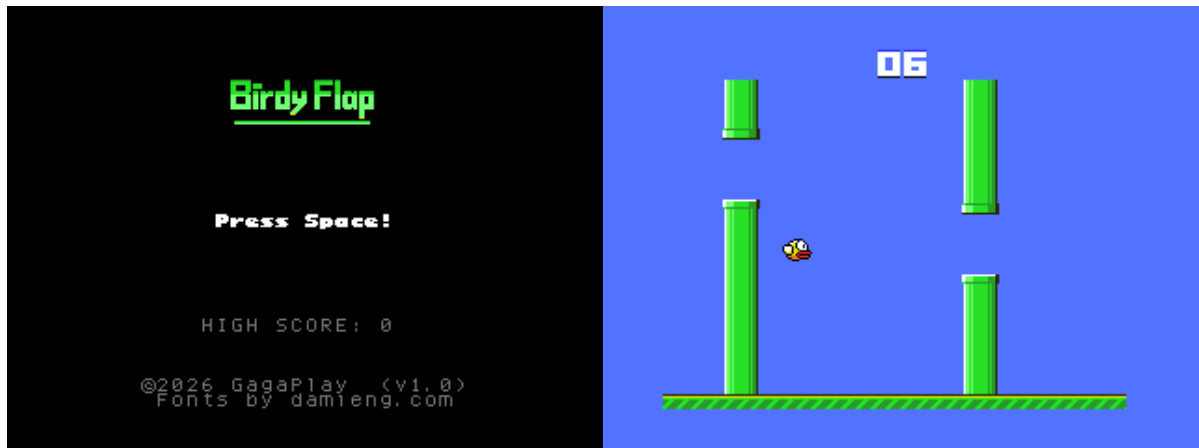


Birdy Flap



A simple Flappy Bird clone for MSX2 and higher, featuring SCC music.

A Konami SCC cartridge is expected in the 2nd cartridge slot for the full music to play.

To run in OpenMSX, enter the following command: `openmsx -extb scc -carta birdyflap.rom`

(NOTE: This assumes the birdyflap.rom file is in your current directory and OpenMSX is in your system path)

Controls

Press SPACE to make the bird flap its wings to move up and avoid the obstacles.

Cheat mode

Press BACK-SPACE during start-up of the MSX until the game's title screen to disable all collision detection.

Credits

- The original Flappy Bird was made by Dong Nguyen (bless this man's soul 😊)
- The MSX adaptation was made by Erik Duijs
 - o The music was inspired by Parodius and Chopin (a variation of the first part of Prelude in E minor)
- Fonts from damieng.com were used

Personal notes

The game itself is obviously just a Flappy Bird clone. Making this started out as just a fun attempt to get back into MSX coding, creating a simple game in assembly and actually finish it.

I had just a simple goal at first: Make it scroll smoothly and make it complete.

In this game however, actually most of my efforts ended up going into creating an efficient and expressive sound/music engine with support for PSG, SCC, and PPI keyboard click.

So 'Birdy Flap' eventually ended up as a vehicle to mostly experiment with this sound engine, which is probably the part of this game that I'm most proud of.

Use of AI

I depended quite a lot on AI to generate code using Claude Sonnet 4.5/4.6, but I did so with very clear instructions and specifications. It's not the result of me simply asking AI to create a Flappy Bird clone with cool music and 'voila, here it is' (very far from it). It was a very deliberate and very iterative process taking small clear steps.

I'd say about 70% of the code was written by AI and 30% was directly written by me. And then out of the generated code, about half of that code was then manually tweaked. None of the graphics or music is generated by AI.

My reasons for using AI:

- I was very rusty in MSX Z80 assembly coding and AI proved to be a very effective springboard to jump back into it.
- Honestly, back in the days (when I was active in FAC) I was never great at Z80 coding to begin with, and this project using AI probably made me a better Z80 coder than I ever was.
- The use of AI was also a practical exercise to see what AI could provide in this sort of scenario, and to optimize a workflow using AI.

The use of AI also brought a lot of frustrations as the generated code was often pretty bad and often outright broke the whole game. AI can be incredibly boneheaded when you know where a bug is, but the AI is trying to convince you that you're wrong and goes on to break something else. AI is not a smooth ride.

However, all things considered the use of AI was a great learning experience.

Anyway, I hope the game is enjoyable.