F’n BALLS

An ZX Spectrum game by Andy Dansby
Loading Screen by Andy Green
Sprite Game Engine Antonio Villena
Programming, Game AI, Block design, music, Sound effects by Andy Dansby
with special help from Antonio Villena, Allen Albright and Ast A. Moore
written with Z88dk and FASE

Works with ZX Spectrum 48k, 128k, +2, +3 Machines

Bob the Ball wants to not be bothered, but alas, this is not Bob’s life, as everyone want to pop Bob. Move Bob to get away from his enemies and find the key to unlock the door to move to the next level.

Bob’s enemies are always seeking him out, they don’t just wander aimlessly. They know where Bob is and will try to find him, so don’t rest too long. All the baddies have to do is touch Bob and you lose a life.

Bob does have a defense. Bob can fire 1 bullet at a time, so aim carefully, as the bullet will travel until it hits an obstacle, an enemy or the edge of the screen.

Bob also needs to avoid various obstacles scattered around the screen.

Skull Block - Kills Bob

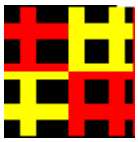
Skull Block - Kills Bob

Enemy Generator – Kills Bob

Why does everything want to kill Bob? Who knows, but there are other things that don’t kill Bob, just annoy Bob.

Stop Block – Stops Bob 

Bounce Block – Bounces Bob in the opposite Direction 

Gate Block – Blocks your Level Block 

There are even some things that help Bob.

Transport Block – Sends Bob to the Lander 

Lander Block – Where Bob gets transported to 

Level Block – Sends Bob to the next level 

Key – Unlock the gate. 

Gives Bob another life. 

Gives Bob some extra points. 

Stop reading. Time to be Bob and avoid those F’n Balls.

Instructions for loading

On a 48k Machine

LOAD “”
press play on your data loader

On a 128k Machine

*do not use 48k mode to load your game, as the game will freeze or otherwise misbehave, use the loader. See notes at the end of the manual

Select **Tape Loader** on your 128k machine on a fresh boot.
Press play on your data loader

Press 1 to select joystick play – Kempston or ZX Interface 2, port 1
Press 2 to select cursor keys or cursor joystick play
Press 3 to select keyboard play – QAOP <Space> keys by default
Press 4 to redefine keys
Press 5 for the High Score Table

Now play the F’n game and have a F’n fun time.

F’n Balls is an open source game, the code which is occasionally updated is at https://github.com/andydansby/fase-bubble

Of course the idea behind FASE is not only to promote the Sprite Engine, but to demonstrate some game concepts such as a simple AI as well as FASE Sprite basics. Do with the code what you will, use the ideas, improve on the game. Give some credit where it’s due, Antonio or to myself, it would be appreciated.

The Sprite Engine is FASE, developed by Antonio Villena, his Github repository is https://github.com/DSkywalk/fase and his product page is at https://www.antoniovillena.es/store/

Antonio Villena assisted me with coding concepts and usage of his FASE engine as well as assisted me with programming tasks. Antonio was also able to resolve a bug with ULA rain happening on real machines (128k Toast Rack and +2 Amstrad Models).

Allen Albright assisted me with Z88dk as well as explaining various concepts, particulars to the ZX Spectrum as well as programming advice in C. Reported issues with ULA Rain.

Andy Green illustrated the loading screen, beta testing and had some great ideas for the game.

Ast. A. Moore assisted with ULA Rain issues on real machines and is a wealth of knowledge.

My Beta testers were George Bachaelor, Krasimir Hristov, Simon Rooney, Marcello Cruz, Andrew Blanche, John Davies, Peter Davidson, Graham Dunn, Chris Wyatt. All who assisted me with testing and debugging.

I want to personally thank everyone who helped me with my silly game.

Thanks for playing F’n Balls.

Notes:

The FASE engine on which this game is built on had some issues originally with generating ULA Rain. This was noticed by Allen Albright, Ast. A Moore, Denis Grachev as well as others. Antonio Villena along with Ast. A Moore was able to fix this particular issue with things that are far above my head, but fix it they did.

48k mode on a 128k machine. Just don’t do it, you will be disappointed, I guarantee it. The game will freeze and bug out, you’ll have to reboot. It might even make you boated and gassy.

The in depth details come from Antonio.

About 48k mode on 128k machines. During the loader, FASE makes a determination on which machine is loading the game

Machine 0: 48K. Engine exploits floating bus (only 48K timings) for anti flickering good case
Machine 1: 128K. Engine exploits shadow video for anti flickering best case
Machine 2: Other. We don't have floating bus or shadow video (detected if paging is activated). Worst case

On machine 0 we have 26k cycles to repaint sprites, so we can manage 8 sprites without flickering
On machine 1 we have a whole frame (69k cycles), so we have more than 12 sprites without flickering
On machine 2 we have only 14k cycles to repaint, that's 4-5 sprites without flickering

The detection algorithm:

```
IF paging
  machine 1
ELSE
  IF bus floating
    machine 0
  ELSE
    machine 2
  END
END
```

The problem when you choose 48k on 128k machine is you have incorrect detection:
-On 128K detects machine 0, so the floating bus algorithm freezes because timings
-On +2A detects machine 2, so no freezes but less sprites without flickering

Machine 2 was designed for other non Sinclair clones like Pentagon, etc.

Truth be told, the game was not designed with clones in mind, so I cannot say if it works properly, hope it does, but if it doesn't, the game is Open Source, so if you fix it, great.