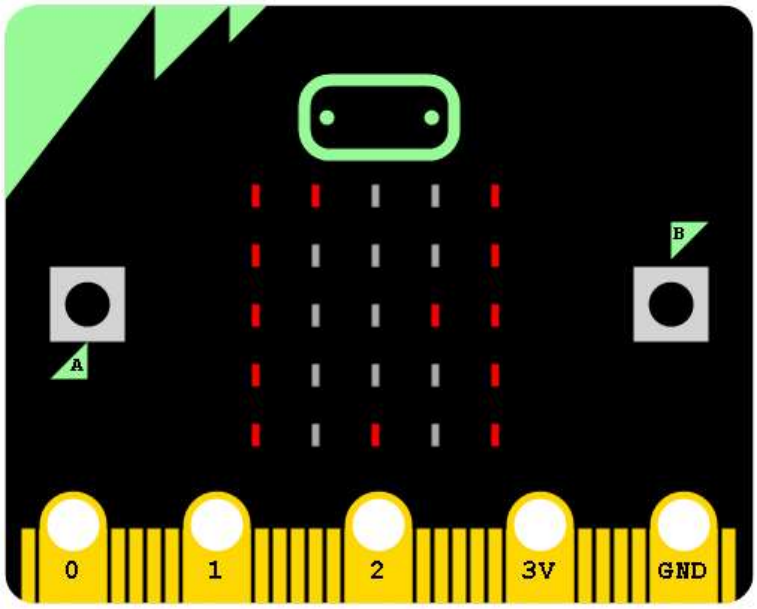
BBC micro:bit
Vertical Scroller

Introduction

Here's another vertical scrolling game. It's a different approach to the scrolling race track. This time, the player can only move to one of 3 cells. The game looks a little like this,



The dot in the centre of the bottom row is the player. The player must avoid oncoming dots in the 3 centre columns of the LED matrix.

Programming

Here is the whole example. Things are pretty basic as they stand but the game is engaging enough in this form to want to work more on it.

```
from microbit import *
import random

def draw(track, t, x, y):
    img = Image(track)
    img.set_pixel(x,y,t * 4 + 1)
    return img

def PlayGame():
    # lines of track
    track_bits = [
        "50005:",
        "53005:",
        "50305:",
        "50035:"
    ]
    # starting track
    track = "50005:50005:50005:50005:50005:"
    # co-ordinates of player character
    x = 2
    y = 4
    # time variable for blinking
    tick = 0
    # track pace
    paws = 75
    last = 0
    alive = True
    while alive==True:
        tt = draw(track,tick,x,y)
        if button_a.was_pressed() and x!=1:
            x = x - 1
        elif button_b.was_pressed() and x!=3:
            x = x + 1
        # update ticks
        tick += 1
        # show track
        display.show(tt)
        # every third tick
        if tick == 3:
            # check for collision
            if tt.get_pixel(x,y-1)!=0:
```

BBC Microbit

Collapse All Expand All

+ Block Editor - The Basics
+ Block Editor - Components
+ Kodu - micro:bit Worlds
+ JavaScript Blocks
+ JavaScript Blocks - Exercises
+ Blocks - Bit:Bot
+ Blocks - Bit:Commander
+ MicroPython - Starting Off
- MicroPython - Examples
  - Dice Rolling
  - Shut The Matrix
  - Encoding Morse Code
  - Encoding Ciphers
  - Drawing A Maze
  - Scrolling Race Track
  - Vertical Scroller
  - Concentration Game
  - Text Entry - Accelerometer
  - Charlie's Python Game
  - Lights Out Game
  - Sam's French Number Game
  - Bryn's Concentrated Clocks
  - Lattice Paths
  - Knight Moves
+ MicroPython - Components
+ MicroPython - Breakout Boards
+ MicroPython - Exercises
+ MicroPython - Pi Accessories
+ MicroPython - Bit:Bot
+ MicroPython - Bit:Commander
+ MicroPython - Projects
+ MicroPython - Visual Basic
+ Other - Odds & Ends

```
                alive = False
            # reset ticks
            tick =0
            # delete bottom row of track
            track = track[:-6]
            # update track
            if last==0:
                last = random.randint(0, len(track_bits)-1)
            else:
                last = 0
            track = track_bits[last] + track
            # redraw screen
            tt = draw(track,tick,x,y)
            display.show(tt)
        sleep(paws)

def main():
    while True:
        if button_a.was_pressed():
            PlayGame()
            sleep(3000)
        sleep(50)

main()
```

Challenges

1. Some tidying up might help. Start with the main function. Display an arrow pointing to the left button and have it blink. This will let the player know how to start. Do some game over images too.
2. Next the game needs a score. Work out a way to keep track of game time. Each tick would be fine. Use this to make a score that you can show to the player at the end of the game.
3. You can decrease the value of the variable **paws** to speed up the game. You could do this when the user picks something up or decrease the value by the same amount after every 100 ticks.
4. Some sound and light effects might be nice. The large pins are enough to give you a couple of lights and a buzzer. You could do a nice 3-2-1 sequence with an F1 style opening.