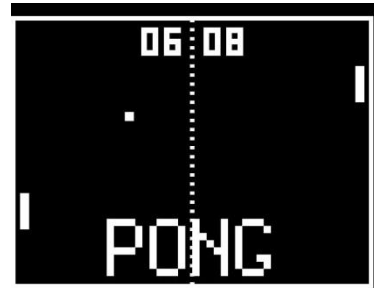


PONG – Add two extra balls



The plan is to allow extra balls to appear when a ball goes out of play on an adjacent computer screen, using the general purpose I/O (GPIO) on the raspberry Pi to communicate. The following instructions allow the extra balls to be added to the game, and also used within the game, before the I/O communication has been implemented.

Because the existing ball is a class it is very easy to create two extra balls by making the following changes (note the ball is initialised with an extra parameter which is the ball number):

```
# Create two bats, three balls and add them to a sprite group
```

```
self.player1Bat = Bat(self.displaySize, "player1")
```

```
self.player2Bat = Bat(self.displaySize, "player2")
```

```
self.ball = Ball(self.displaySize, 0)
```

```
self.ball2 = Ball(self.displaySize, 1)
```

```
self.ball3 = Ball(self.displaySize, 2)
```

```
self.sprites = sprite.Group(self.player1Bat, self.player2Bat,
```

```
self.ball, self.ball2, self.ball3)
```

And now we need to replicate the code that is used to test for the original ball collisions so that collisions are detected for the two new balls:

```
# Check for bat collisions
```

```
self.ball.batCollisionTest(self.player1Bat)
```

```
self.ball.batCollisionTest(self.player2Bat)
```

```
# Check for bat collisions (ball2)
```

```
self.ball2.batCollisionTest(self.player1Bat)
```

```
self.ball2.batCollisionTest(self.player2Bat)
```

```
# Check for bat collisions (ball3)
```

```
self.ball3.batCollisionTest(self.player1Bat)
```

```
self.ball3.batCollisionTest(self.player2Bat)
```

P.T.O...

The behaviour of the extra balls is different from the original because they are sometimes visible and sometimes not visible. They should appear on request (either a key press or an input from another Pi) but then disappear when they go out of play. They also need to be a different colour. Firstly change the `__init__` function for the ball:

```
# The class for the ball

class Ball(sprite.Sprite):

    def __init__(self, displaySize, ballnumber):
```

Still within the `__init__` function make the following changes to store the ball number and also change the colour of the ball:

```
        self.ballnumber = ballnumber
        # Fill the image to colour the ball
        if ballnumber == 2:
            self.image.fill((0, 0, 255)) # blue
        elif ballnumber == 1:
            self.image.fill((255, 0, 0)) # red
        else:
            self.image.fill((255, 255, 255)) # white
```

P.T.O...

Now change the ball's reset function so that the extra balls start off hidden:

```
def reset(self):  
  
    if (self.ballnumber == 0): # the main ball, always showing  
        self.showing = 1  
        # Start the ball directly in the centre of the screen  
        self.rect.centerx = self.displaySize[0] / 2  
        self.rect.centery = self.displaySize[1] / 2  
  
        # Start the ball moving to the left or right (pick randomly)  
        # Vector(x, y)  
        if random.randrange(1, 3) == 1:  
            # move to left  
            self.vector = (-1, 0)  
        else:  
            # move to right  
            self.vector = (1, 0)  
    else: # other balls start off hidden  
        self.showing = 0  
        # Put the ball off the screen  
        self.rect.centerx = -100  
        self.rect.centery = -100  
  
        # Stop the ball moving  
        # Vector(x, y)  
        self.vector = (0, 0)
```

P.T.O...

Immediately after this add a new show function for the ball (needed for the extra balls). Note that one of the extra balls always comes from the left and the other always comes from the right:

```
def show(self):
    if (self.showing == 0):
        # Start the ball directly in the centre of the screen
        self.rect.centerx = self.displaySize[0] / 2
        self.rect.centery = self.displaySize[1] / 2
        if self.ballnumber == 2:
            # move to left
            self.vector = (-1, 0)
        else: # ball number 1
            # move to right
            self.vector = (1, 0)
        self.showing = 1
```

Now insert this one line at the start of the ball Update function. Any ball that is not visible does not need to be updated:

```
def update(self):
    if self.showing == 1: # no update if the ball is hidden
```

And similarly for the batCollisionTest function:

```
def batCollisionTest (self):
    if self.showing == 1: # no collision test needed if the ball is hidden
```

All that is required now is to show the extra balls when a key is pressed. “B” for ball 2 and “N” for ball 3. Add this code to the HandleEvents function in the main game class:

```
if event.key == K_ESCAPE:
    pygame.quit()
    sys.exit()

if event.key == K_b: # make ball 2 appear
    self.ball2.show()

if event.key == K_n: # make ball 3 appear
    self.ball3.show()
```