Pygame lesson 6

In this tutorial I will introduce you to collisions. You can do this without any fancy coding at all – just keep track of the x and y coordinates of your objects. No problem if there are a couple of characters on the screen, but a bit hassle if there are loads of things.

The Pygame Sprite class makes this easy. In this example, I have made Ball class and based it upon the Pygame Sprite class. Here are some highlights of the code:

```
class Ball(pygame.sprite.Sprite):
    """Class to keep track of a ball's location and vector."""
    def __init__(self):
        super().__init__()
        self.size=randrange(20,100)

        self.x = randrange(self.size, SCREEN_WIDTH - self.size)
        self.y = randrange(self.size, SCREEN_HEIGHT - self.size)
        self.change_x = randrange(-3, 3)
        self.change_y = randrange(-3, 3)

        self.color=(randrange(0,255), randrange(0,255), randrange(0,255))

        self.rect = pygame.Rect(0,0,0,0)
```

The super().__init__ means to create the Ball sprite based upon the Pygame.sprite.Spite() class, inheriting all its properties. There are loads of properties:

https://www.pygame.org/docs/ref/sprite.html#pygame.sprite.Sprite

I then created the sprites and put them in a group. Groups will be more useful when I'm not just animating two things on the screen…

```
# create individual balls and assign to Group
balls = pygame.sprite.Group()
ball1 = Ball()
ball2 = Ball()
balls.add(ball1)
balls.add(ball2)
```

In the game loop I took care of the collision. The 'bouncing' effect is pretty silly, but there are various things you might want to do, this just gives an idea:

```
    if ball1.rect.colliderect(ball2.rect):
        print(ball1.rect, ball2.rect)
        ball1.change_x *= -1
        ball1.change_y *= -1
        ball2.change_x *= -1
        ball2.change_y *= -1
```

Below is all the code. You should run it and then tinker a bit to make sure you understand it properly.

```
# Sources:
# http://programarcadegames.com/
# https://realpython.com/pygame-a-primer/
# https://www.pygame.org/docs/ref/sprite.html
# https://www.pygame.org/project-Rect+Collision+Response-1061-.html

import pygame
from random import randrange
from sys import exit

# Define some colors
BLACK = (0, 0, 0)
WHITE = (255, 255, 255)

SCREEN_WIDTH = 700
SCREEN_HEIGHT = 500

class Ball(pygame.sprite.Sprite):
    """Class to keep track of a ball's location and vector."""
    def __init__(self):
        super().__init__()
        self.size=randrange(20,100)

        self.x = randrange(self.size, SCREEN_WIDTH - self.size)
        self.y = randrange(self.size, SCREEN_HEIGHT - self.size)
        self.change_x = randrange(-3, 3)
        self.change_y = randrange(-3, 3)

        self.color=(randrange(0,255), randrange(0,255), randrange(0,255))

        self.rect = pygame.Rect(0,0,0,0)

# main program
pygame.init()

# Set the height and width of the screen
size = [SCREEN_WIDTH, SCREEN_HEIGHT]
screen = pygame.display.set_mode(size)

pygame.display.set_caption("Bouncing Balls")

# Loop until the user clicks the close button.
done = False

# Used to manage how fast the screen updates
clock = pygame.time.Clock()

# create individual balls and assign to Group
balls = pygame.sprite.Group()
ball1 = Ball()
```

```
ball2 = Ball()
balls.add(ball1)
balls.add(ball2)

# -------- Main Program Loop -----------
while not done:
    # --- Event Processing
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done = True

    # --- Logic
    ball1.x += ball1.change_x
    ball1.y += ball1.change_y
    ball2.x += ball2.change_x
    ball2.y += ball2.change_y

    ball1.rect = pygame.Rect(ball1.x-ball1.size,ball1.y-ball1.size,ball1.size*2,
ball1.size*2)
    ball2.rect = pygame.Rect(ball2.x-ball1.size,ball2.y-ball2.size,ball2.size*2,
ball2.size*2)
    #print(ball1.rect, ball2.rect)

    # Bounce ball1 if needed
    if ball1.y > SCREEN_HEIGHT - ball1.size or ball1.y < ball1.size:
        ball1.change_y *= -1
    if ball1.x > SCREEN_WIDTH - ball1.size or ball1.x < ball1.size:
        ball1.change_x *= -1

    # Bounce ball2 if needed
    if ball2.y > SCREEN_HEIGHT - ball2.size or ball2.y < ball2.size:
        ball2.change_y *= -1
    if ball2.x > SCREEN_WIDTH - ball2.size or ball2.x < ball2.size:
        ball2.change_x *= -1

    if ball1.rect.colliderect(ball2.rect):
        print(ball1.rect, ball2.rect)
        ball1.change_x *= -1
        ball1.change_y *= -1
        ball2.change_x *= -1
        ball2.change_y *= -1

    # --- Drawing
    # Set the screen background
    screen.fill(BLACK)

    # Draw the balls
    for ball in balls:
        pygame.draw.circle(screen, ball.color, [ball.x, ball.y], ball.size)
```

```
    # --- Wrap-up
    clock.tick(60)

    # Go ahead and update the screen with what we've drawn.
    pygame.display.flip()

# Close everything down
pygame.quit()
```