Banana Bash

In this exercise, we'll develop a 2D game similar to Pizza Panic, called Banana Bash. A screenshot of the game is shown below.

The object of the game is to catch the falling bananas while avoiding the coconuts. For each banana the monkey catches, the player gets 10 points. If a banana falls without being caught or if a coconut hits the monkey, the game ends.

Assume the following components are available: If you want any other images in your implementation, you will need to create these.

- background.gif -- the background image shown
- bananas.gif -- an image containing the bananas
- monkey.gif -- an image containing the monkey
- coconut.gif -- an image containing the coconut

1) Design classes for the monkey, the bananas, and the coconut. Specify what the base class for these will be and what methods are overridden (if any).
2) How will you implement the falling coconuts and bananas? *Hint:* The way this was done in Pizza Panic was to have a chef sprite randomly traverse the top of the screen and drop falling pizzas. Can we use a similar approach here?
3) Implement this game. You will probably find the code for Pizza Panic useful, so it is given on the next pages.
from livewires import games, color
import random

games.init(screen_width = 640, screen_height = 480, fps = 50)

class Pan(games.Sprite):
    """
    A pan controlled by player to catch falling pizzas.
    """
    image = games.load_image("pan.bmp")

    def __init__(self):
        """
        Initialize Pan object and create Text object for score.
        """
        super(Pan, self).__init__(image = Pan.image,
                                  x = games.mouse.x,
                                  bottom = games.screen.height)

        self.score = games.Text(value = 0, size = 25, color = color.black,
                                 top = 5, right = games.screen.width - 10)

        games.screen.add(self.score)

    def update(self):
        """
        Move to mouse x position.
        """
        self.x = games.mouse.x

        if self.left < 0:
            self.left = 0

        if self.right > games.screen.width:
            self.right = games.screen.width

        self.check_catch()

    def check_catch(self):
        """
        Check if catch pizzas.
        """
        for pizza in self.overlapping_sprites:
            self.score.value += 10
            self.score.right = games.screen.width - 10
            pizza.handle_caught()

class Pizza(games.Sprite):
    """
    A pizza which falls to the ground.
    """
    image = games.load_image("pizza.bmp")
    speed = 1

    def __init__(self, x, y = 90):
        """
        Initialize a Pizza object.
        """
        super(Pizza, self).__init__(image = Pizza.image,
                                     x = x, y = y,
                                     dy = Pizza.speed)

    def update(self):
        """
        Check if bottom edge has reached screen bottom.
        """
        if self.bottom > games.screen.height:
            self.end_game()
            self.destroy()

    def handle_caught(self):
        """
        Destroy self if caught.
        """
        self.destroy()
def end_game(self):
    """ End the game. """
    end_message = games.Message(value = "Game Over",
                                size = 90,
                                color = color.red,
                                x = games.screen.width/2,
                                y = games.screen.height/2,
                                lifetime = 5 * games.screen.fps,
                                after_death = games.screen.quit)
    games.screen.add(end_message)

class Chef(games.Sprite):
    """
    A chef which moves left and right, dropping pizzas.
    """
    image = games.load_image("chef.bmp")

    def __init__(self, y = 55, speed = 2, odds_change = 200):
        """ Initialize the Chef object. """
        super(Chef, self).__init__(image = Chef.image,
                                    x = games.screen.width / 2,
                                    y = y,
                                    dx = speed)
        self.odds_change = odds_change
        self.time_til_drop = 0

    def update(self):
        """ Determine if direction needs to be reversed. """
        if self.left < 0 or self.right > games.screen.width:
            self.dx = -self.dx
        elif random.randrange(self.odds_change) == 0:
            self.odds_change = -self.odds_change
        self.check_drop()

    def check_drop(self):
        """ Decrease countdown or drop pizza and reset countdown. """
        if self.time_til_drop > 0:
            self.time_til_drop -= 1
        else:
            new_pizza = Pizza(x = self.x)
            games.screen.add(new_pizza)

            # set buffer to approx 30% of pizza height, regardless of pizza speed
            self.time_til_drop = int(new_pizza.height * 1.3 / Pizza.speed) + 1

def main():
    """ Play the game. """
    wall_image = games.load_image("wall.jpg", transparent = False)
    games.screen.background = wall_image

    the_chef = Chef()
    games.screen.add(the_chef)

    the_pan = Pan()
    games.screen.add(the_pan)

    games.mouse.is_visible = False
    games.screen.event_grab = True
    games.screen.mainloop()

    # start it up!
main()