Original

from tkinter import \*

import random

import time

class Ball:

def \_\_init\_\_(self, canvas, color):

self.canvas = canvas

self.id = canvas.create\_oval(10, 10, 25, 25, fill=color)

self.canvas.move(self.id, 245, 100)

starts = [-3, -2, -1, 1, 2, 3]

self.x = random.choice(starts)

self.y = -3

self.canvas\_height = self.canvas.winfo\_height()

self.canvas\_width = self.canvas.winfo\_width()

def draw(self):

self.canvas.move(self.id, self.x, self.y)

pos = self.canvas.coords(self.id)

if pos[0] <= 0 or pos[2] >= self.canvas\_width:

self.x = self.x \* -1

if pos[1] <= 0 or pos[3] >= self.canvas\_height:

self.y = self.y \* -1

if \_\_name\_\_ == '\_\_main\_\_':

tk = Tk()

tk.title('Bounce Game')

tk.resizable(True, True)

tk.wm\_attributes('-topmost', True)

canvas = Canvas(tk, width=500, height=400, bd=0, highlightthickness=10)

canvas.pack()

tk.update()

ball = Ball(canvas, 'red')

while True:

ball.draw()

tk.update\_idletasks()

tk.update()

time.sleep(0.01)

tk.mainloop()

from tkinter import \*

import random

import time

class Ball:

def \_\_init\_\_(self, canvas, color):

self.canvas = canvas

self.id = canvas.create\_oval(10, 10, 25, 25, fill=color)

self.canvas.move(self.id, 245, 100)

self.colors = ["red", "blue", "white", "black", "purple"]

posibleStartingPostion = [-3, -2, -1, 1, 2, 3]

# if xDirectionAndSpeed is negative, the ball goes right to left

# if xDirectionAndSpeed is positive, the ball goes left to right

# if xDirectionAndSpeed is bigger, the ball moves faster

self.xDirectionAndSpeed = random.choice(posibleStartingPostion)

self.yDirectionAndSpeed = -3

self.canvas\_height = self.canvas.winfo\_height()

self.canvas\_width = self.canvas.winfo\_width()

def moveBall(self):

# move the ball

self.canvas.move(self.id, self.xDirectionAndSpeed, self.yDirectionAndSpeed)

# get the coordinates of the ball

pos = self.canvas.coords(self.id)

# if the left side of the ball hits the left wall

# or the right side of the ball hits the right wall

# invert the direction of the ball

if pos[0] <= 0 or pos[2] >= self.canvas\_width:

self.xDirectionAndSpeed = self.xDirectionAndSpeed \* -1.05

canvas.itemconfig(self.id, fill=random.choice(self.colors))

# similar

if pos[1] <= 0 or pos[3] >= self.canvas\_height:

self.yDirectionAndSpeed = self.yDirectionAndSpeed \* -1.05

canvas.itemconfig(self.id, fill=random.choice(self.colors))

if \_\_name\_\_ == "\_\_main\_\_":

tk = Tk()

tk.title('Bounce Game')

tk.resizable(0, 0)

tk.wm\_attributes('-topmost', 1)

canvas = Canvas(tk, width=500, height=400, bd=0, highlightthickness=0)

canvas.pack()

tk.update()

ball = Ball(canvas, "green")

while True:

ball.moveBall()

tk.update\_idletasks()

tk.update()

time.sleep(0.01)

# oval = canvas.create\_oval(10, 10, 25, 25, fill="red")

# canvas.move(oval, 100, 0)

tk.mainloop()

from tkinter import \*

import random

import time

class Ball:

def \_\_init\_\_(self, canvas, color):

self.canvas = canvas

self.id = canvas.create\_oval(10, 10, 25, 25, fill=color)

self.canvas.move(self.id, 245, 100)

starts = [-3, -2, -1, 1, 2, 3]

self.x = random.choice(starts)

self.y = -3

self.canvas\_height = self.canvas.winfo\_height()

self.canvas\_width = self.canvas.winfo\_width()

def draw(self):

self.canvas.move(self.id, self.x, self.y)

pos = self.canvas.coords(self.id)

if pos[0] <= 0 or pos[2] >= self.canvas\_width:

self.x = self.x \* -1

if pos[1] <= 0 or pos[3] >= self.canvas\_height:

self.y = self.y \* -1

if \_\_name\_\_ == "\_\_main\_\_":

tk = Tk()

tk.title('Bounce Game')

tk.resizable(0, 0)

tk.wm\_attributes('-topmost', 1)

canvas = Canvas(tk, width=500, height=400, bd=0, highlightthickness=0)

canvas.pack()

tk.update()

ball = Ball(canvas, 'red')

while True:

ball.draw()

tk.update\_idletasks()

tk.update()

time.sleep(0.01)

from tkinter import \*

import random

import time

class Ball:

def \_\_init\_\_(self, canvas, color):

self.canvas = canvas

self.id = canvas.create\_oval(10, 10, 25, 25, fill=color)

self.canvas.move(self.id, 245, 100)

posibleStartingPostion = [-3, -2, -1, 1, 2, 3]

# if xDirectionAndSpeed is negative, the ball goes right to left

# if xDirectionAndSpeed is positive, the ball goes left to right

# if xDirectionAndSpeed is bigger, the ball moves faster

self.xDirectionAndSpeed = random.choice(posibleStartingPostion)

self.yDirectionAndSpeed = -3

self.canvas\_height = self.canvas.winfo\_height()

self.canvas\_width = self.canvas.winfo\_width()

def moveBall(self):

# move the ball

self.canvas.move(self.id, self.xDirectionAndSpeed, self.yDirectionAndSpeed)

# get the coordinates of the ball

pos = self.canvas.coords(self.id)

# if the left side of the ball hits the left wall

# or the right side of the ball hits the right wall

# invert the direction of the ball

if pos[0] <= 0 or pos[2] >= self.canvas\_width:

self.xDirectionAndSpeed = self.xDirectionAndSpeed \* -1.05

# similar

if pos[1] <= 0 or pos[3] >= self.canvas\_height:

self.yDirectionAndSpeed = self.yDirectionAndSpeed \* -1.05

if \_\_name\_\_ == "\_\_main\_\_":

tk = Tk()

tk.title('Bounce Game')

tk.resizable(0, 0)

tk.wm\_attributes('-topmost', 1)

canvas = Canvas(tk, width=500, height=400, bd=0, highlightthickness=0)

canvas.pack()

tk.update()

ball = Ball(canvas, "green")

while True:

ball.moveBall()

tk.update\_idletasks()

tk.update()

time.sleep(0.01)

# oval = canvas.create\_oval(10, 10, 25, 25, fill="red")

# canvas.move(oval, 100, 0)

tk.mainloop()