import pygame

# -- Global constants

# Colors

BLACK = (0, 0, 0)

WHITE = (255, 255, 255)

BLUE = (50, 50, 255)

# Screen dimensions

SCREEN\_WIDTH = 800

SCREEN\_HEIGHT = 600

class Player(pygame.sprite.Sprite):

""" This class represents the bar at the bottom that the player

controls. """

# Constructor function

def \_\_init\_\_(self, x, y):

# Call the parent's constructor

super().\_\_init\_\_()

# Set height, width

self.image = pygame.Surface([15, 15])

self.image.fill(WHITE)

# Make our top-left corner the passed-in location.

self.rect = self.image.get\_rect()

self.rect.y = y

self.rect.x = x

# Set speed vector

self.change\_x = 0

self.change\_y = 0

self.walls = None

def changespeed(self, x, y):

""" Change the speed of the player. """

self.change\_x += x

self.change\_y += y

def update(self):

""" Update the player position. """

# Move left/right

self.rect.x += self.change\_x

# Did this update cause us to hit a wall?

block\_hit\_list = pygame.sprite.spritecollide(self, self.walls, False)

for block in block\_hit\_list:

# If we are moving right, set our right side to the left side of

# the item we hit

if self.change\_x > 0:

self.rect.right = block.rect.left

else:

# Otherwise if we are moving left, do the opposite.

self.rect.left = block.rect.right

# Move up/down

self.rect.y += self.change\_y

# Check and see if we hit anything

block\_hit\_list = pygame.sprite.spritecollide(self, self.walls, False)

for block in block\_hit\_list:

# Reset our position based on the top/bottom of the object.

if self.change\_y > 0:

self.rect.bottom = block.rect.top

else:

self.rect.top = block.rect.bottom

class Wall(pygame.sprite.Sprite):

""" Wall the player can run into. """

def \_\_init\_\_(self, x, y, width, height):

""" Constructor for the wall that the player can run into. """

# Call the parent's constructor

super().\_\_init\_\_()

# Make a blue wall, of the size specified in the parameters

self.image = pygame.Surface([width, height])

self.image.fill(BLUE)

# Make our top-left corner the passed-in location.

self.rect = self.image.get\_rect()

self.rect.y = y

self.rect.x = x

# Call this function so the Pygame library can initialize itself

pygame.init()

# Create an 800x600 sized screen

screen = pygame.display.set\_mode([SCREEN\_WIDTH, SCREEN\_HEIGHT])

# Set the title of the window

pygame.display.set\_caption('Test')

# List to hold all the sprites

all\_sprite\_list = pygame.sprite.Group()

# Make the walls. (x\_pos, y\_pos, width, height)

wall\_list = pygame.sprite.Group()

wall = Wall(0, 0, 10, 600)

wall\_list.add(wall)

all\_sprite\_list.add(wall)

wall = Wall(10, 0, 790, 10)

wall\_list.add(wall)

all\_sprite\_list.add(wall)

wall = Wall(10, 200, 100, 10)

wall\_list.add(wall)

all\_sprite\_list.add(wall)

# Create the player paddle object

player = Player(50, 50)

player.walls = wall\_list

all\_sprite\_list.add(player)

clock = pygame.time.Clock()

done = False

while not done:

for event in pygame.event.get():

if event.type == pygame.QUIT:

done = True

elif event.type == pygame.KEYDOWN:

if event.key == pygame.K\_LEFT:

player.changespeed(-3, 0)

elif event.key == pygame.K\_RIGHT:

player.changespeed(3, 0)

elif event.key == pygame.K\_UP:

player.changespeed(0, -3)

elif event.key == pygame.K\_DOWN:

player.changespeed(0, 3)

elif event.type == pygame.KEYUP:

if event.key == pygame.K\_LEFT:

player.changespeed(3, 0)

elif event.key == pygame.K\_RIGHT:

player.changespeed(-3, 0)

elif event.key == pygame.K\_UP:

player.changespeed(0, 3)

elif event.key == pygame.K\_DOWN:

player.changespeed(0, -3)

all\_sprite\_list.update()

screen.fill(BLACK)

all\_sprite\_list.draw(screen)

pygame.display.flip()

clock.tick(60)

pygame.quit()