# program template for Spaceship

import simplegui

import math

import random

# globals for user interface

WIDTH = 800

HEIGHT = 600

score = 0

lives = 3

time = 0

ANGLE\_VEL\_INC = 0.02

Acceleration = 0.1

Friction = 0.01

class ImageInfo:

def \_\_init\_\_(self, center, size, radius = 0, lifespan = None, animated = False):

self.center = center

self.size = size

self.radius = radius

if lifespan:

self.lifespan = lifespan

else:

self.lifespan = float('inf')

self.animated = animated

def get\_center(self):

return self.center

def get\_size(self):

return self.size

def get\_radius(self):

return self.radius

def get\_lifespan(self):

return self.lifespan

def get\_animated(self):

return self.animated

# art assets created by Kim Lathrop, may be freely re-used in non-commercial projects, please credit Kim

# debris images - debris1\_brown.png, debris2\_brown.png, debris3\_brown.png, debris4\_brown.png背景

# debris1\_blue.png, debris2\_blue.png, debris3\_blue.png, debris4\_blue.png, debris\_blend.png

debris\_info = ImageInfo([320, 240], [640, 480])

debris\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/debris2\_blue.png")

# nebula images - nebula\_brown.png, nebula\_blue.png背景

nebula\_info = ImageInfo([400, 300], [800, 600])

nebula\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/nebula\_blue.f2014.png")

# splash image開始畫面

splash\_info = ImageInfo([200, 150], [400, 300])

splash\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/splash.png")

# ship image飛行船

ship\_info = ImageInfo([45, 45], [90, 90], 35)

ship\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/double\_ship.png")

# missile image - shot1.png, shot2.png, shot3.png子彈

missile\_info = ImageInfo([5,5], [10, 10], 3, 50)

missile\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/shot2.png")

# asteroid images - asteroid\_blue.png, asteroid\_brown.png, asteroid\_blend.png石頭

asteroid\_info = ImageInfo([45, 45], [90, 90], 40)

asteroid\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/asteroid\_blue.png")

# animated explosion - explosion\_orange.png, explosion\_blue.png, explosion\_blue2.png, explosion\_alpha.png

explosion\_info = ImageInfo([64, 64], [128, 128], 17, 24, True)

explosion\_image = simplegui.load\_image("http://commondatastorage.googleapis.com/codeskulptor-assets/lathrop/explosion\_alpha.png")

# sound assets purchased from sounddogs.com, please do not redistribute

soundtrack = simplegui.load\_sound("http://commondatastorage.googleapis.com/codeskulptor-assets/sounddogs/soundtrack.mp3")

missile\_sound = simplegui.load\_sound("http://commondatastorage.googleapis.com/codeskulptor-assets/sounddogs/missile.mp3")

missile\_sound.set\_volume(.5)

ship\_thrust\_sound = simplegui.load\_sound("http://commondatastorage.googleapis.com/codeskulptor-assets/sounddogs/thrust.mp3")

explosion\_sound = simplegui.load\_sound("http://commondatastorage.googleapis.com/codeskulptor-assets/sounddogs/explosion.mp3")

# alternative upbeat soundtrack by composer and former IIPP student Emiel Stopler

# please do not redistribute without permission from Emiel at http://www.filmcomposer.nl

#soundtrack = simplegui.load\_sound("https://storage.googleapis.com/codeskulptor-assets/ricerocks\_theme.mp3")

# helper functions to handle transformations

def angle\_to\_vector(ang):

return [math.cos(ang), math.sin(ang)]

def dist(p,q):

return math.sqrt((p[0] - q[0]) \*\* 2+(p[1] - q[1]) \*\* 2)

# Ship class

class Ship:

def \_\_init\_\_(self, pos, vel, angle, image, info):

self.pos = [pos[0],pos[1]]

self.vel = [vel[0],vel[1]]

self.thrust = False

self.angle = angle

self.angle\_vel = 0

self.image = image

self.image\_center = info.get\_center()

self.image\_size = info.get\_size()

self.radius = info.get\_radius()

def draw(self,canvas):

canvas.draw\_image(self.image, self.image\_center, self.image\_size, self.pos, self.image\_size, self.angle)

def update(self):

# update ship angle

self.angle += self.angle\_vel

self.pos[0] += self.vel[0]

self.pos[1] += self.vel[1]

# ship thrust Acceleration and Friction

vel = angle\_to\_vector(self.angle)

if self.thrust == True:

self.vel[0] += Acceleration \* vel[0]

self.vel[1] += Acceleration \* vel[1]

else:

self.vel[0] \*= (1-Friction)

self.vel[1] \*= (1-Friction)

#outo of canvas

if self.pos[0] + 45 > WIDTH:

self.pos[0] = 0

if self.pos[1] + 45 > HEIGHT:

self.pos[1] = 0

if self.pos[0] < 0:

self.pos[0] = WIDTH - 45

if self.pos[1] < 0:

self.pos[1] = HEIGHT - 45

def shoot(self, missile):

vel = angle\_to\_vector(self.angle)

missile.pos = [self.pos[0] + 45\*vel[0] , self.pos[1] + 45\*vel[1]]

missile.vel = [self.vel[0] + vel[0]\*1.5 , self.vel[1] + vel[1]\*1.5]

missile.sound.play()

# Sprite class

class Sprite:

def \_\_init\_\_(self, pos, vel, ang, ang\_vel, image, info, sound = None):

self.pos = [pos[0],pos[1]]

self.vel = [vel[0],vel[1]]

self.angle = ang

self.angle\_vel = ang\_vel

self.image = image

self.image\_center = info.get\_center()

self.image\_size = info.get\_size()

self.radius = info.get\_radius()

self.lifespan = info.get\_lifespan()

self.animated = info.get\_animated()

self.age = 0

self.sound = sound

if sound:

sound.rewind()

sound.play()

def draw(self, canvas):

canvas.draw\_image(self.image, self.image\_center, self.image\_size, self.pos, self.image\_size, self.angle)

def update(self):

self.angle += self.angle\_vel

self.pos[0] += self.vel[0]

self.pos[1] += self.vel[1]

# define keyhandlers to control firing\_angle

def keydown(key):

if simplegui.KEY\_MAP["space"] == key:

my\_ship.shoot(a\_missile)

elif simplegui.KEY\_MAP["up"] == key:

my\_ship.thrust = True

ship\_thrust\_sound.play()

elif simplegui.KEY\_MAP["left"] == key:

my\_ship.angle\_vel -= ANGLE\_VEL\_INC

elif simplegui.KEY\_MAP["right"] == key:

my\_ship.angle\_vel += ANGLE\_VEL\_INC

def keyup(key):

global my\_ship, ANGLE\_VEL\_INC, ship\_thrust\_sound

if simplegui.KEY\_MAP["up"] == key:

my\_ship.thrust = False

ship\_thrust\_sound.rewind()

elif simplegui.KEY\_MAP["left"] == key:

my\_ship.angle\_vel += ANGLE\_VEL\_INC

elif simplegui.KEY\_MAP["right"] == key:

my\_ship.angle\_vel -= ANGLE\_VEL\_INC

def draw(canvas):

global time

# animiate background

time += 1

wtime = (time / 4) % WIDTH

center = debris\_info.get\_center()

size = debris\_info.get\_size()

canvas.draw\_image(nebula\_image, nebula\_info.get\_center(), nebula\_info.get\_size(), [WIDTH / 2, HEIGHT / 2], [WIDTH, HEIGHT])

canvas.draw\_image(debris\_image, center, size, (wtime - WIDTH / 2, HEIGHT / 2), (WIDTH, HEIGHT))

canvas.draw\_image(debris\_image, center, size, (wtime + WIDTH / 2, HEIGHT / 2), (WIDTH, HEIGHT))

# ship thrust image

if my\_ship.thrust == False:

my\_ship.image\_center = [45, 45]

else:

my\_ship.image\_center = [90+45, 45]

# draw ship and sprites

my\_ship.draw(canvas)

a\_rock.draw(canvas)

a\_missile.draw(canvas)

# missile position

if a\_missile.pos[0] + 45 > WIDTH:

a\_missile.pos[0] = 0

if a\_missile.pos[1] + 45 > HEIGHT:

a\_missile.pos[1] = 0

if a\_missile.pos[0] < 0:

a\_missile.pos[0] = WIDTH - 45

if a\_missile.pos[1] < 0:

a\_missile.pos[1] = HEIGHT - 45

# update ship and sprites

my\_ship.update()

a\_rock.update()

a\_missile.update()

# Update score, lives

canvas.draw\_text('Lives' , (50, 50), 30, 'white', 'monospace')

canvas.draw\_text(str(lives), (50, 90), 30, 'white', 'sans-serif')

canvas.draw\_text('Score' , (WIDTH - 150, 50), 30, 'white', 'monospace')

canvas.draw\_text(str(score), (WIDTH - 150, 90), 30, 'white', 'sans-serif')

# timer handler that spawns a rock

def rock\_spawner():

global a\_rock

a\_rock.pos = [random.choice(range(0, WIDTH)), random.choice(range(0, HEIGHT))]

a\_rock.angle\_vel = random.random() / 50

a\_rock.angle += random.choice([1, -1]) \* a\_rock.angle\_vel

# initialize frame

frame = simplegui.create\_frame("Asteroids", WIDTH, HEIGHT)

# initialize ship and two sprites

my\_ship = Ship([WIDTH / 2, HEIGHT / 2], [0, 0], 0, ship\_image, ship\_info)

a\_rock = Sprite([WIDTH / 3, HEIGHT / 3], [0, 0], 0, 0.05, asteroid\_image, asteroid\_info)

a\_missile = Sprite([2 \* WIDTH / 3, 2 \* HEIGHT / 3], [-1,1], 0, 0, missile\_image, missile\_info, missile\_sound)

# register handlers

frame.set\_draw\_handler(draw)

timer = simplegui.create\_timer(1000.0, rock\_spawner)

# control

frame.set\_keydown\_handler(keydown)

frame.set\_keyup\_handler(keyup)

# get things rolling

timer.start()

frame.start()