A Varaint of the Snake Game

20211030 박정원

https://github.com/jewerlycider/Jungwon20211030.github.io.git

```
import pygame
import os
import sys
import random
from time import sleep

# 게임 스크린 전역변수
SCREEN_WIDTH = 800
SCREEN_HEIGHT = 600

# 게임 화면 전역변수
GRID_SIZE = 20
GRID_WIDTH = SCREEN_WIDTH / GRID_SIZE
GRID_HEIGHT = SCREEN_HEIGHT / GRID_SIZE

# 방향 전역변수
UP = (0, -1)
DOWN = (0, 1)
LEFT = (-1, 0)
RIGHT = (1, 0)

# 색상 전역변수
WHITE = (255, 255, 255)
ORANGE = (250, 150, 0)
GRAY = (100, 100, 100)
BLUE = (0,102,204)
BLACK = (0,0,0)

#배경음악 재생
pygame.init()
pygame.mixer.init()
pygame.mixer.music.load('/Users/parkjungwon/Desktop/visual media
programming/assignments_code/Ghostrifter Official - Afternoon Nap.mp3')
pygame.mixer.music.play(-1) #-1 --> 무한반복

# 뱀 객체
class Snake(object):
    def __init__(self):
        self.create()

    # 뱀 생성
    def create(self):
        self.length = 2
        self.positions = [(int(SCREEN_WIDTH / 2), int(SCREEN_HEIGHT / 2))]
```

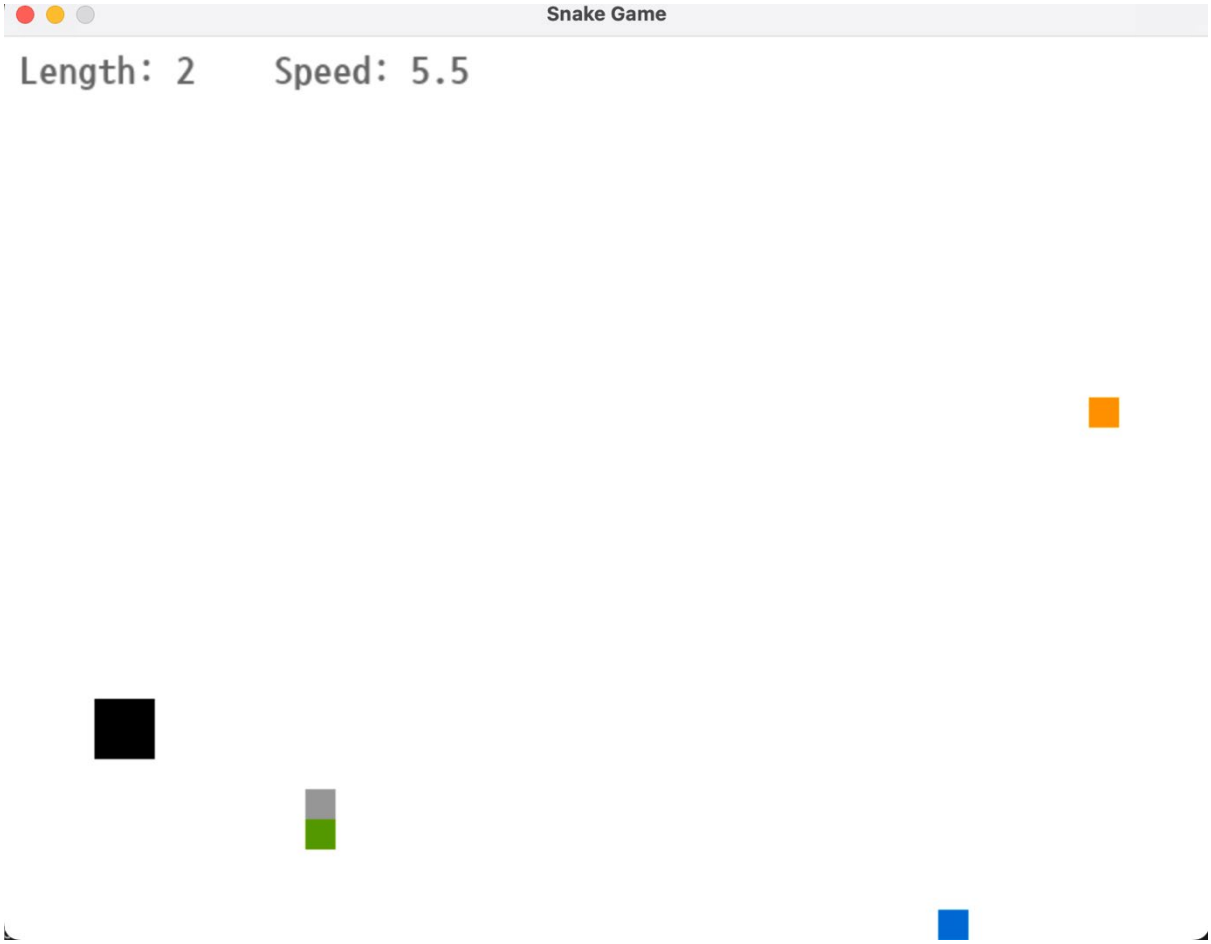
```
            self.direction = random.choice([UP, DOWN, LEFT, RIGHT])

    # 뱀 방향 조정
    def control(self, xy):
            if (xy[0] * -1, xy[1] * -1) == self.direction:
                    return
            else:
                    self.direction = xy

    # 뱀 이동
    def move(self):
            cur = self.positions[0]
            x, y = self.direction
            new = (cur[0] + (x * GRID_SIZE)), (cur[1] + (y * GRID_SIZE))

            # 뱀이 자기 몸통에 닿았을 경우 뱀 처음부터 다시 생성
            if new in self.positions[2:]:
                    sleep(1)
                    self.create()
            # 뱀이 게임화면을 넘어갈 경우 뱀 처음부터 다시 생성
            elif new[0] < 0 or new[0] >= SCREEN_WIDTH or \
                            new[1] < 0 or new[1] >= SCREEN_HEIGHT:
                    sleep(1)
                    self.create()
            # 뱀이 정상적으로 이동하는 경우
            else:
                    self.positions.insert(0, new)
                    if len(self.positions) > self.length:
                            self.positions.pop()

    # 뱀이 먹이를 먹을 때 호출
    def eat(self):
            self.length += 1

    # 뱀 그리기
    def draw(self, screen):
            red, green, blue = 50 / (self.length - 1), 150, 150 / (self.length - 1)
            red2,green2,blue2 = 250, 50 / (self.length - 1), 150 / (self.length - 1)
            red3,green3,blue3 = 50 / (self.length - 1), 150 / (self.length - 1), 230
            for i, p in enumerate(self.positions):
                    if self.length>=10:
                            color = (red2, 100+green2*i, blue2 * i)
                    elif self.length>=30:
                            color = (100 + red3 * i, green3*i, blue3)
                    else:
                            color = (100 + red * i, green, blue * i)
                    rect = pygame.Rect((p[0], p[1]), (GRID_SIZE, GRID_SIZE))
                    pygame.draw.rect(screen, color, rect)

#장애물 객체
```

```
class Obstacle(object):
    def __init__(self):
        self.position = (0,0)
        self.color = BLACK
        self.create()

    def create(self):
        x = random.randint(0, GRID_WIDTH - 1)
        y = random.randint(0, GRID_HEIGHT - 1)
        self.position = x * GRID_SIZE, y * GRID_SIZE

    def draw(self, screen):
        o_rect = pygame.Rect((self.position[0], self.position[1]), (GRID_SIZE*2, GRID_SIZE*2)) #먹이의
2 배크기
        pygame.draw.rect(screen, self.color, o_rect)


# 먹이 객체
class Feed(object):
    def __init__(self):
        self.position1 = (0, 0)
        self.color1 = ORANGE
        self.position2 = (0, 0)
        self.color2 = BLUE
        self.create1()
        self.create2()

    # 먹이 생성
    def create1(self):
        x1 = random.randint(0, GRID_WIDTH - 1)
        y1 = random.randint(0, GRID_HEIGHT - 1)
        self.position1 = x1 * GRID_SIZE, y1 * GRID_SIZE

    def create2(self):
        x2 = random.randint(0, GRID_WIDTH - 1)
        y2 = random.randint(0, GRID_HEIGHT - 1)
        self.position2 = x2 * GRID_SIZE, y2 * GRID_SIZE

    # 먹이 그리기
    def draw(self, screen):
        rect1 = pygame.Rect((self.position1[0], self.position1[1]), (GRID_SIZE, GRID_SIZE))
        rect2 = pygame.Rect((self.position2[0], self.position2[1]), (GRID_SIZE, GRID_SIZE))
        pygame.draw.rect(screen, self.color1, rect1)
        pygame.draw.rect(screen, self.color2, rect2)


# 게임 객체
class Game(object):
    def __init__(self):
        self.snake = Snake()
```

```
        self.feed = Feed()
        self.obstacle = Obstacle()
        self.speed = 20

    # 게임 이벤트 처리 및 조작
    def process_events(self):
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                return True
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_UP:
                    self.snake.control(UP)
                elif event.key == pygame.K_DOWN:
                    self.snake.control(DOWN)
                elif event.key == pygame.K_LEFT:
                    self.snake.control(LEFT)
                elif event.key == pygame.K_RIGHT:
                    self.snake.control(RIGHT)
        return False

    # 게임 로직 수행
    def run_logic(self):
        self.snake.move()
        self.check_eat(self.snake, self.feed,self.obstacle)
        self.check_obstacle(self.snake, self.obstacle)
        self.speed = (20 + self.snake.length) / 4

    #뱀이 장애물에 부딪혔는지 체크
    def check_obstacle(self,snake,obstacle):
        if snake.positions[0] == (obstacle.position):
            sleep(1)
            snake.create()
        elif snake.positions[0] == (obstacle.position[0]+GRID_SIZE,obstacle.position[1]):
            sleep(1)
            snake.create()
        elif snake.positions[0] == (obstacle.position[0],obstacle.position[1]+GRID_SIZE):
            sleep(1)
            snake.create()
        elif snake.positions[0] == (obstacle.position[0]+GRID_SIZE,obstacle.position[1]+GRID_SIZE):
            sleep(1)
            snake.create()

    # 뱀이 먹이를 먹었는지 체크
    def check_eat(self, snake, feed,obstacle):
        if snake.positions[0] == (feed.position1):
            snake.eat()
            feed.create1()
            obstacle.create() #먹이 먹으면 장애물 다시 생성

        if snake.positions[0] == (feed.position2):
```

```
                    snake.eat()
                    feed.create2()
                    obstacle.create()


        def resource_path(self, relative_path):
                try:
                        base_path = sys._MEIPASS
                except Exception:
                        base_path = os.path.abspath(".")
                return os.path.join(base_path, relative_path)

        # 게임 정보 출력
        def draw_info(self, length, speed, screen):
                info = "Length: " + str(length) + "        " + "Speed: " + str(round(speed, 2))
                font_path = resource_path("/Users/parkjungwon/Desktop/visual media
programming/assignments_code/NanumGothicCoding-Bold.ttf")
                font = pygame.font.Font(font_path, 26)
                text_obj = font.render(info, 1, GRAY)
                text_rect = text_obj.get_rect()
                text_rect.x, text_rect.y = 10, 10
                screen.blit(text_obj, text_rect)

        # 게임 프레임 처리
        def display_frame(self, screen):
                screen.fill(WHITE)
                self.draw_info(self.snake.length, self.speed, screen)
                self.snake.draw(screen)
                self.feed.draw(screen)
                self.obstacle.draw(screen)
                screen.blit(screen, (0, 0))

# 리소스 경로 설정
def resource_path(relative_path):
        try:
                base_path = sys._MEIPASS
        except Exception:
                base_path = os.path.abspath(".")
        return os.path.join(base_path, relative_path)


def main():
        # 게임 초기화 및 환경 설정
        pygame.init()
        pygame.display.set_caption('Snake Game')
        screen = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))


        clock = pygame.time.Clock()
        game = Game()
```

```
    done = False
    while not done:
        done = game.process_events()
        game.run_logic()
        game.display_frame(screen)
        pygame.display.flip()
        clock.tick(game.speed)


    pygame.quit()


if __name__ == '__main__':
    main()
```


Snake Game
Length: 2
Speed: 5.5


Snake Game
Length: 2
Speed: 5.5