

Pygame1

20221024민정현

Github link: https://github.com/jeonghyeoni/snake_game_variation

I added background music, sound effects, new feeds, levels, and score recording functions in the existing snake game.

* For ease of explanation, the order is different from the actual code.

1. Code Explanation

1) Background music and sound effects

```
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()
    current_path = os.path.dirname(__file__)
    assets_path = os.path.join(current_path, 'assets')
    pygame.mixer.music.load(os.path.join(assets_path, '8bit_music.mp3'))
    pygame.mixer.music.play(-1) # 무한반복
    pygame.mixer.music.set_volume(0.2) # 배경음악 볼륨 조절
    sound = pygame.mixer.Sound(os.path.join(assets_path, 'sound.mp3'))
    sound.set_volume(0.2) # 효과음 볼륨 조절
    game = Game(sound) # 뱀이 먹이를 먹으면 효과음이 나오도록 하기 위해
                        # sound를 Game class의 parameter로 넣어 줌

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

    pygame.quit()

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

```
class Game(object):
    def __init__(self, sound):
        self.snake = Snake()
        self.feed = Feed()
        self.speed = 5
        self.original_speed = 5
        self.sound = sound    #sound 를 객체 변수로 정의
        self.level = 1
```

```
# 게임 로직 수행
def run_logic(self):
    self.snake.move()
    self.if_died_init_setting()
    self.check_eat(self.snake, self.feed) # 먹이를 먹었는지 확인하는 함수
    self.speed = (20 + self.snake.length) / 4 + self.feed.extraSpeed()
    self.original_speed = (20 + self.snake.length) / 4
    self.create_feed(self.snake, self.feed)
```

```
# 뱀이 먹이를 먹었는지 확인
def check_eat(self, snake, feed):
    if snake.positions[0] in feed.positions:
        snake.eat()
        self.sound.play() # 뱀 머리 위치가 먹이 위치 리스트 안에 있으면 효과음 재생
        index = feed.positions.index(snake.positions[0])
        feed.last_color = feed.colors[index]
        feed.last_speed = feed.extra_speed
```

2) New feeds that increase or decrease speed additionally when eaten by snake.

```
# Game class
# 게임 로직 수행
def run_logic(self):
    self.snake.move()
    self.if_died_init_setting()
    self.check_eat(self.snake, self.feed)
    self.speed = (20 + self.snake.length) / 4 + self.feed.extraSpeed()
    self.original_speed = (20 + self.snake.length) / 4
    self.create_feed(self.snake, self.feed) # 먹이를 생성하는 함수
```

```

# 뱀이 먹이를 먹었으면 새로운 위치에 먹이 생성
def create_feed(self, snake, feed):
    # 먹이 위치 리스트에 뱀 머리 위치가 존재할 시
    if snake.positions[0] in feed.positions:
        self.check_level()
        index = feed.positions.index(snake.positions[0])
        feed.positions.remove(snake.positions[0])
        feed.colors.pop(index)
        feed.create() # Feed class 의 create() 함수를 불러옴

```

```

class Feed(object):
    def __init__(self):
        self.positions = []
        self.colors = [ORANGE] # 처음 색은 주황색
        self.last_color = ORANGE
        self.last_speed = 0
        self.extra_speed = 0
        self.n = 3 # 초기 먹이의 개수
        self.create() # 게임 시작시 먹이 3 개가 바로 생성 됨

    # 먹이 생성
    def create(self):
        for i in range(self.n): # 먹이 개수 만큼 반복
            x = random.randint(0, GRID_WIDTH - 1) # 좌표 랜덤 형성
            y = random.randint(0, GRID_HEIGHT - 1)
            self.extra_speed = self.last_speed
            # 먹이 위치 리스트에 좌표 추가
            self.positions.append((x * GRID_SIZE, y * GRID_SIZE))
            self.n = 1 # 새로 추가될 먹이 개수를 1 개로 바꿈

```

```

# main
while not done:
    done = game.process_events()
    game.run_logic()
    game.display_frame(screen) # 화면에 모든 요소를 표시
    pygame.display.flip()
    clock.tick(game.speed)

pygame.quit()

```



```
def create_feed(self, snake, feed):  
    if snake.positions[0] in feed.positions:  
        self.check_level() <<<<<<<<<<<<<<<<<<<<<<<<  
        index = feed.positions.index(snake.positions[0])  
        feed.positions.remove(snake.positions[0])  
        feed.colors.pop(index)  
        feed.create()
```

```
def check_level(self):
    self.last_level = self.level # last_level 에 현재 level 저장
    self.level = self.snake.length//10 + 1 #level 업데이트
    if self.last_level < self.level: # level up 했을 시 추가되는 먹이 개수
        self.feed.n = 3                # 3 개로 지정
```

[illegible]

```
#Feed class
def create(self):
    for i in range(self.n):
        x = random.randint(0, GRID_WIDTH - 1)
        y = random.randint(0, GRID_HEIGHT - 1)
        self.extra_speed = self.last_speed
        self.positions.append((x * GRID_SIZE, y * GRID_SIZE))
        self.n = 1 # level up 한 뒤 먹이를 처음 먹었을 때만 새로운 먹이가 3 개
                    # 추가되고 그 뒤에 먹이를 먹을 때는 평소대로 1 개만 추가 됨
```

4) Score recording

```
class Snake(object):
    def __init__(self):
        self.best_length = 2 # 초기 best_length 2로 설정
        self.create()
```

create함수가 아닌 init에서 best_length를 정의한 이유는 create함수에서 정의할 시 뱀이 죽고 다시 생성될 때 best_length가 초기화되기 때문이다.

```
def eat(self):
    self.length += 1
    if self.length > self.best_length:
        self.best_length = self.length
```

먹이를 먹을 때마다 뱀의 length가 1씩 더해지고, 만약 현재 length가 best_length보다 크다면 (최고기록 갱신) best_length를 현재 length로 업데이트한다.

```
#Game class #best_length를 화면에 표시해야 의미있음
def draw_info(self, level, length, speed, screen, best_length):
    info = "Level " + str(level) + " " + "Length: " + str(length) +
        " " + "Speed: " + str(round(speed, 2)) + " " + " " +
        str(round(self.feed.extraSpeed(), 2))
    # 기존 코드에서 level과 extra speed도 표시되도록 수정
    font_path = resource_path("assets/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)

    # 최고 기록은 따로 텍스트를 만들어 표시
    info2 = 'Best Length: ' + str(best_length)
    if self.snake.length == self.snake.best_length:
        font_color = RED #현재 최고기록 갱신 중이면 빨간색으로 표시
    else:
        font_color = GRAY # 최고기록 갱신 전이면 회색으로 표시
    text_obj2 = font.render(info2, 1, font_color)
    text_rect2 = text_obj2.get_rect()
    text_rect2.x, text_rect2.y = SCREEN_WIDTH-250, 10
    screen.blit(text_obj2, text_rect2)
```

2. Screen Views

초기화면



초록색 먹이를 먹었을 때 속도 감소



길이가 10이 되었을 때 level up(먹이 3개 추가)



Game over 됐을 때 best length 기록, 먹이 개수 초기화, 레벨 초기화 (이 사진에서 레벨 초기화가 안 된 것을 보고 코드 수정 완료 했습니다.)

