창의 소프트웨어 기말 과제

Python으로 게임 만들어 보기

201521033 조경선

[1] 개요

본 과제를 시작하기 앞서, 본인은 '자료구조'라는 수업에서 이미 Python을 가지고 프로그래밍을 해 본 경험이 있다. 그래서 이번 과제가 자유 주제인 만큼 해보지 못했던 것과 특별한 것을 해보고 싶었다. 인터넷 서칭을 해본 결과 Pygame이라는 module을 사용하면 손쉽게 간단한 게임을 만들 수 있다고 보았다. 그래서 본인은 Pygame을 통해 일명 '짱구붕어빵 게임'을 만들었다. 짱구붕어빵 게임은 붕어빵을 좋아하는 짱구가 붕어빵이 내뱉는 팥을 먹는 게임이다. 3번의 기회가 있는데 팥을 3번 놓치면 게임은 끝나게 된다. 단순하지만 가벼운 마음으로 구현하기에 쉬울 것 같아 이를 창의소프트웨어 기말과제로 준비했다.

[2] 계획

일단 Pygame이 어떻게 구성이 되는지, 함수들은 어떤 것이 있는지 살펴보아야 했다. 그래서 본인은 Pygame basics 사이트에서 기본적으로 어떻게 프로그래밍을 하는지 차근차근 배워갔다. 게임의 콘솔창을 설정하는 일이 먼저였기에 그림 1과 같이 코딩을 해보았다.

```
1. import pygame, sys
2. from pygame.locals import *
3.
4. pygame.init()
5. DISPLAYSURF = pygame.display.set_mode((400, 300))
6. pygame.display.set_caption('Hello World!')
7. while True: # main game loop
8.     for event in pygame.event.get():
9.         if event.type == QUIT:
10.            pygame.quit()
11.            sys.exit()
12.    pygame.display.update()
```

When you run this program, a black window like this will appear:

그림 1 게임 콘솔창 설정 (출처: http://inventwithpython.com/pygame/chapter2.html#)

이렇게 기본적인 세팅을 한 후, 게임을 어떻게 만들어 갈 것인지 단계별로 정리해보았다.

1. 기본적인 전역변수들 설정: 색깔, 초당 프레임, 이미지들의 크기, 짱구, 붕어빵, 그리고 팥의 위

치 초기화

2. Txt_dis(text) 함수: 문자를 화면에 출력시켜주는 함수
3. Miss(count): 남은 생명 수를 출력시켜주는 함수
4. Gameover(): 생명이 하나도 안 남았을 때 게임을 끝내 주는 함수
5. Draw_object(obj, x, y): 이미지를 콘솔창에 출력시켜주는 함수
6. Init_bg(): 오프닝을 위해 배경을 초기화 시켜주는 함수
7. Run_Game(): 게임이 진행될 때 모든 일들이 진행되는 함수 - key event, quit event, 붕어빵의 움직임, 팥의 움직임, 초당 프레임 설정 모든 것이 포함
8. Init_Game(): 오프닝 게임 콘솔을 토기화 시켜주는 함수
9. 마지막으로 init_Game() 함수와 run_Game() 함수 호출

이 계획과 함수들을 세우기 위해 “Python을 이용한 간단한 슈팅게임 만들기” 블로그를 참고하였다.

[3] 실행

전역변수까지는 기본적인 변수들을 설정하고, txt_dis(text)에서는 pygame의 스타일로 구현해야 했다. Font 설정도 가능했고, 글자의 위치도 설정을 직접 해 주었다. 그림2와 같이 구현하였다.

```
#display text
def txt_dis(text):
    global Background
    txtfont = pygame.font.Font(None, 50)
    text_ob = txtfont.render(text, True, BLACK)
    textpos = text_ob.get_rect()
    textpos.center = (bg_width/2, bg_height/2)
    Background.blit(text_ob, textpos)
    pygame.display.update()
```

그림 2 txt_dis(text) 함수

다음으로 miss와 gameover 함수 모두 text들을 출력해주는 함수가 주가 되었고, gameover 함수에서는 게임에서 졌기 때문에 게임이 종료되는 방향으로 프로그래밍을 하였다. Draw_obect는 말그대로 이미지를 화면에 출력해주는 명령어를 넣어주었고, init_bg 함수에서느 이미지들을 로드하고 이를 출력해주는 함수를 구현하였다. 그리고 그림 3이 init_bg함수를 구현한 것인데, 마지막 부분에는 본인의 이름과 학번을 왼쪽 아래에 넣어 주기 위해 텍스트 출력함수를 추가하였다.

가장 중요한 함수인 init_Game()은 게임이 진행되고 있을 때 계속 돌아야 하는 함수이다. 일단 key event를 설정해주기 위해 처음에는 event.type == KEYDOWN인지 확인한 후 짱구의 위치를 바꿔주었다. 하지만 이 때 발생한 문제는 키를 한번 누를 때마다 실행이 되고, 키를 누르고 있을 때 연속적으로

움직이지 않았다. 그래서 pygame.key.get_pressed()를 이용하여 키를 누를고 있을 때 연속적으로 짱구가 움직이도록 설정하였다. 이는 그림 4에서 보는 것과 같다.

```
def init_bg():
    global Background, kid2_img, fish2_img, heart_img
    Background.fill(WHITE)
    kid2_img = pygame.image.load('kid2.png')
    fish2_img = pygame.image.load('fish2.png')
    heart_img = pygame.image.load('heart.png')
    draw_object(kid2_img, bg_width*0.55, bg_height*0.1)
    draw_object(heart_img, 250, bg_height*0.1)
    draw_object(fish2_img, 300, bg_height*0.17)

    name_font = pygame.font.SysFont(None, 20)
    name_text = name_font.render('Chokyungsun 201521033', True, BLACK)
    Background.blit(name_text, (10, 530))
```

그림 3 init_bg() 함수

```
#for pressing key event
keystate = pygame.key.get_pressed()
if keystate[K_DOWN]:
    kid_ychange += 8
if keystate[K_UP]:
    kid_ychange -= 8
if keystate[K_RIGHT]:
    kid_xchange += 8
if keystate[K_LEFT]:
    kid_xchange -= 8

for event in pygame.event.get():
    if event.type == QUIT:
        pygame.quit()
        sys.exit()
    if event.type == KEYDOWN:
        if event.key == K_DOWN:
            kid_ychange += 8
        if event.key == K_UP:
            kid_ychange -= 8
        if event.key == K_RIGHT:
            kid_xchange += 8
        if event.key == K_LEFT:
            kid_xchange -= 8
```

그림 4 key event

그 다음 짱구의 위치, 붕어빵의 위치와 움직임, 그리고 팥의 위치와 짱구와의 충돌검사를 모두 시켜주고 그 다음으로 gameover를 확인시켜주고 모든 생명이 떨어지지 않았으면 run_Game함수를 다시 불

러들인다. 마지막으로 init_Game 함수는 게임을 시작하기 전에 설정해줘야 하는 모든 것들을 구현해주었다.

[4] 테스트 결과

테스트를 해본 결과는 그림 5와 6과 같다. 오프닝에서는 처음에 게임 이름이 나오고 3초를 세고 시작을 한다. 시간상 복잡한 게임은 만들지 못했지만 짱구는 중간에 있는 선 안에서만 움직일 수 있고, 팥이 날라들 때 먹지 못하면 생명수가 하나씩 줄어든다. 붕어빵은 팥을 주기적으로 내뱉고 있다.

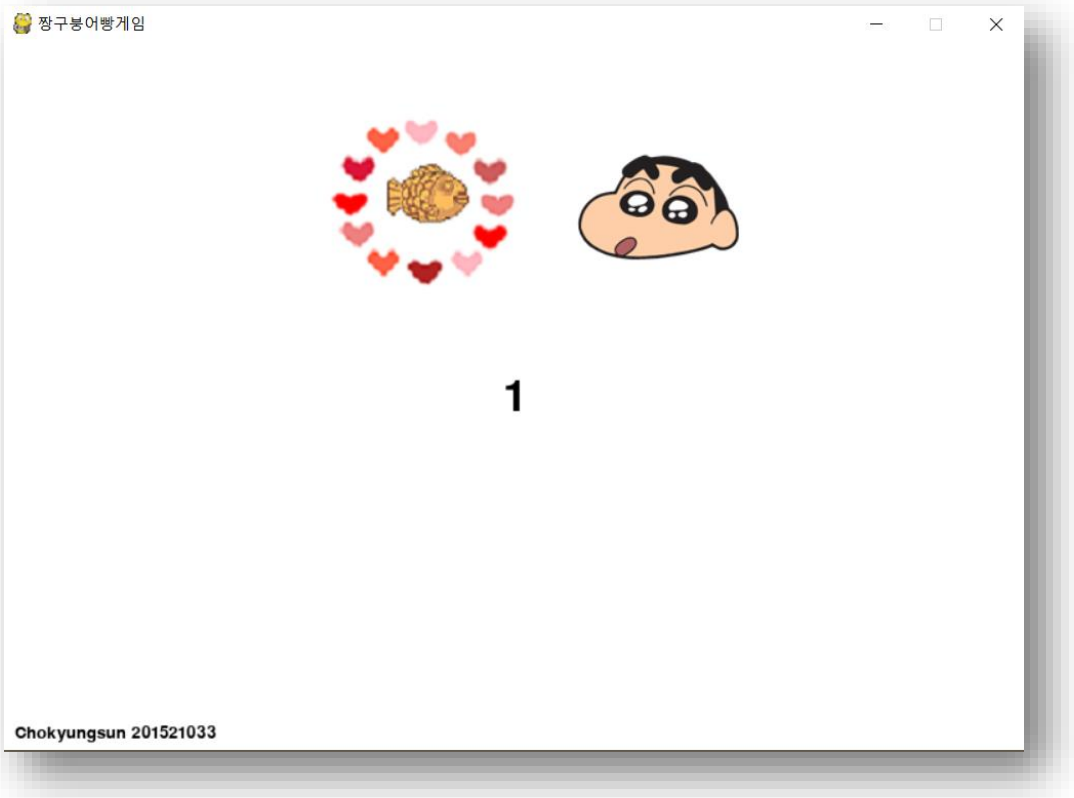

그림 5 오프닝 장면

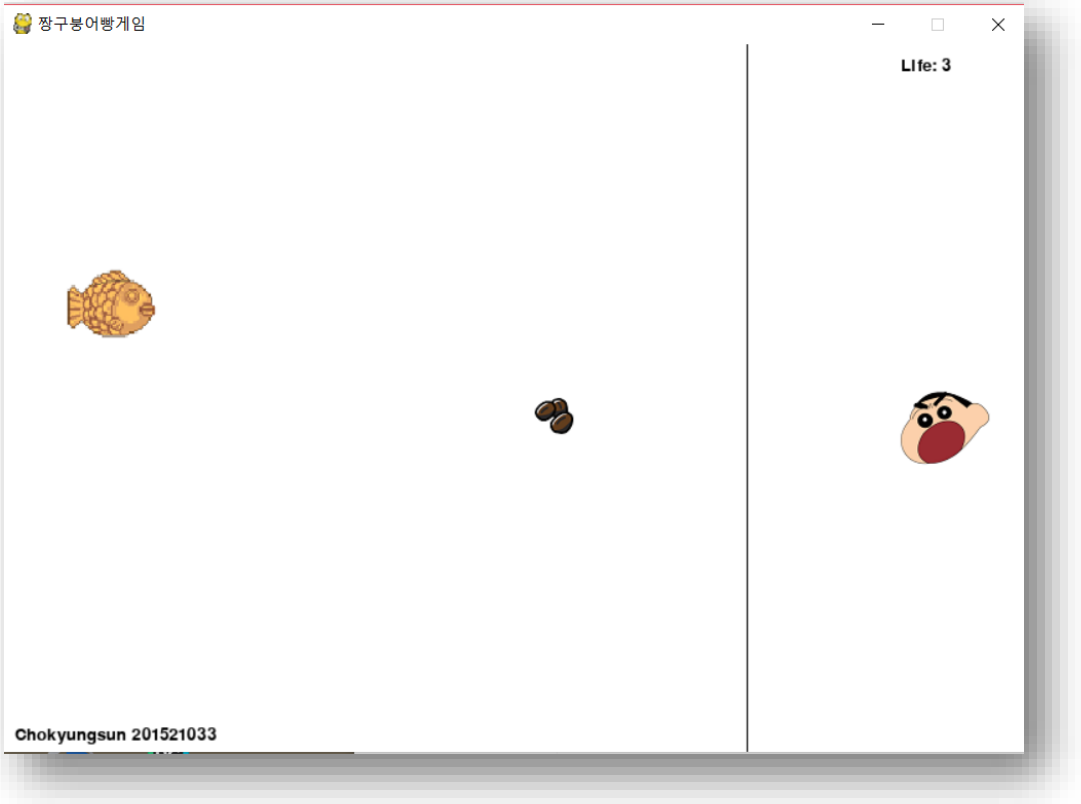

그림 6 게임 진행 중

[5] 결론

이번 과제를 진행하면서 혼자 아무것도 모르는 모듈을 가지고 프로젝트를 진행해본 것은 처음이었다. 파이썬으로 게임을 만들 수 있다는 것을 아예 몰랐을 뿐 더러, 이번 프로젝트를 진행하면서 '게임프로그래밍' 수업이 많이 생각이 났다. 게임프로그래밍 수업은 C언어로 진행하는 수업이었는데, C언어로 게임을 만드는 것보다 훨씬 함수 구현도 쉬웠고, 더 나아가 win32에서 구현했던 게임들의 코딩 방식이 비슷해서 이번 과제에서도 손쉽게 이해하며 구현할 수 있었다.

다만 아쉬운 점은 시간이 많이 없어 정교하게 만들지 못했다는 점이다. 기본적으로 게임 오버가 되는 규칙뿐 만 아니라 어떻게 하면 게임에서 이길 수 있을지에 대한 규칙도 있어야 했는데 이는 시간 부족으로 구현하지 못했다. 예를 들어, 60초 동안 팥을 일정량을 먹어야 한다던가, 혹은 시간이 갈수록 점수가 점점 올라가 max 점수를 찍는 것에 유저가 기쁨을 느끼게 한다는 등의 구체적인 게임 룰이 들어가지 못해서 아쉽다. 다음에는 더 정교하고 UI적으로도 퀄리티가 좋은 게임을 만들 수 있을 것이라 기대한다.

[6] 참고 문헌

"Python을 이용한 간단한 슈팅게임 만들기", http://lidron.tistory.com/41

"Chapter2 - Pygame Basics", http://inventwithpython.com/pygame/chapter2.html