

Pygame으로 게임만들기 4

7. 적과 충돌하기

```
4 import pygame
5 import sys
6 import random
7 import time
8 from pygame.locals import *
9
10 # 상수영역
11 # 초당 프레임 수
12 FPS = 30
13 # 윈도우 크기, 비율 일정하게 만듦
14 WINDOWWIDTH = 1080
15 WINDOWHEIGHT = int(WINDOWWIDTH * 0.75)
16 # 배경 최대 크기
17 ORIGINBACKGROUNDWIDTH = 1280
18 ORIGINBACKGROUNDHEIGHT = 640
19 # 배경 속도
20 BACKGROUNDSPEED = 2
21 # 색
22 WHITE = (255, 255, 255)
23 RED = (255, 0, 0)
24
25
26 class AirPlane(pygame.sprite.Sprite):
27     def __init__(self):
28         self.image = pygame.image.load('airplane.png')
29         self.rect = self.image.get_rect()
30         self.rect.left = WINDOWWIDTH * 0.05
31         self.rect.top = WINDOWHEIGHT * 0.8
32
33     def change_y(self, value):
34         if self.rect.top + value < 0:
35             self.rect.top = 0
36         elif self.rect.top + value > WINDOWHEIGHT - self.image.get_height():
37             self.rect.top = WINDOWHEIGHT - self.image.get_height()
38         else:
39             self.rect.top += value
40
41     def change_x(self, value):
42         if self.rect.left + value < 0:
43             self.rect.left = 0
44         elif self.rect.left + value > WINDOWWIDTH - self.image.get_width():
45             self.rect.left = WINDOWWIDTH - self.image.get_width()
46         else:
47             self.rect.left += value
48
49     def position(self):
50         return self.rect.left, self.rect.top
```

7. 적과 충돌하기

```
59 class FireBall(pygame.sprite.Sprite):
60     FIREBALLSPEED = 15
61     FIREBALLNUM = 1
62     PROBABILITY = 7
63
64     def __init__(self):
65         global IMAGESDICT
66         super().__init__()
67         self.fireball_choice = random.randint(1, self.PROBABILITY)
68         if self.fireball_choice <= 2:
69             self.image = IMAGESDICT["fireball%s" % self.fireball_choice]
70             self.rect = self.image.get_rect()
71             self.rect.left, self.rect.top = init_enemy_pos(self.image)
72         else:
73             self.image = IMAGESDICT["blank"]
74             self.rect = self.image.get_rect()
75             self.rect.left = WINDOWWIDTH
76             self.rect.top = -2
77
78     def update(self):
79         if self.rect.left <= 0:
80             self.kill()
81         if self.fireball_choice <= 2:
82             self.rect.left -= self.FIREBALLSPEED
83         else:
84             self.rect.left -= 2 * self.FIREBALLSPEED
```

7. 적과 충돌하기

```
class Boom(pygame.sprite.Sprite):
    BOOMTIME = 5

    def __init__(self, x, y):
        global IMAGESDICT
        super().__init__()
        self.image = IMAGESDICT["boom"]
        self.rect = self.image.get_rect()
        self.rect.left = x
        self.rect.top = y
        self.time = 0

    def update(self):
        self.time += 1
        if self.time >= self.BOOMTIME:
            self.kill()

class BatEnemy(pygame.sprite.Sprite):
    BATSPEED = 7
    BATTIME = 3
    bat_num = 0
    bat_remove_time = 0

    def __init__(self):
        global IMAGESDICT
        super().__init__()
        self.image = IMAGESDICT["bat"]
        self.rect = self.image.get_rect()
        self.rect.left, self.rect.top = init_enemy_pos(self.image)
        BatEnemy.bat_num += 1

    def __del__(self):
        BatEnemy.bat_num -= 1
        BatEnemy.bat_remove_time = time.time()

    def update(self):
        self.rect = self.rect.move(-self.BATSPEED, 0)
        if self.rect.left < 0:
            self.kill()

    def position(self):
        return self.rect.left, self.rect.top
```

7. 적과 충돌하기

```
176 class AirplaneBullet(pygame.sprite.Sprite):
177     """
178     """
179     BULLETSPEED = 15
180
181     def __init__(self, airplane_xy):
182         """
183         """
184         global IMAGESDICT
185         super().__init__()
186         self.image = IMAGESDICT["bullet"]
187         self.rect = self.image.get_rect()
188         self.rect.left = airplane_xy[0] + IMAGESDICT["airplane"].get_width()
189         self.rect.top = airplane_xy[1] + IMAGESDICT["airplane"].get_height() / 2
190
191     def update(self):
192         """
193         """
194         self.rect = self.rect.move(self.BULLETSPEED, 0)
195         if self.rect.left > WINDOWWIDTH:
196             self.kill()
197
198 def text_obj(text, font, color):
199     """
200     """
201     text_surface = font.render(text, True, color)
202     return text_surface, text_surface.get_rect()
203
204 def disp_message(sentence, pos_x, pos_y, size, color):
205     """
206     """
207     text = pygame.font.Font('freesansbold.ttf', int(size*WINDOWWIDTH/ORIGINBACKGROUNDWIDTH))
208     text_surf, text_rect = text_obj(sentence, text, color)
209     text_rect.center = (pos_x, pos_y)
210     draw_object(text_surf, text_rect.left, text_rect.top)
```

7. 적과 충돌하기

```
def crash():
    disp_message("Crashed!", WINDOWWIDTH/2, WINDOWHEIGHT/2, 115, RED)
    pygame.display.update()
    pygame.time.delay(2000)
    main()

def init_enemy_pos(image):
    x = WINDOWWIDTH
    y = random.randrange(0, WINDOWHEIGHT - image.get_height())
    return x, y

def draw_object(image, x, y):
    global DISPLAYSURF
    DISPLAYSURF.blit(image, (x, y))

def main():
    global FPSLOCK, DISPLAYSURF
    global IMAGESDICT

    # 비행기 왼쪽 초기 위치
    airplane_y_change = 0
    airplane_x_change = 0

    # 윈도우 변경에 따른 배경크기 변경
    BACKGROUNDWIDTH = IMAGESDICT["background"].get_width()

    # 배경 초기 위치
    background_x = 0
    other_background_x = BACKGROUNDWIDTH

    # 개별 sprite group
    bullet_group = pygame.sprite.Group()
    bat_group = pygame.sprite.Group()
    boom_group = pygame.sprite.Group()
    fireball_group = pygame.sprite.Group()
    # 전체 sprite group
    sprite_group = pygame.sprite.Group()

    # 비행기 sprite
    airplane = AirPlane()
    sprite_group.add(airplane)
```

7. 적과 충돌하기

```
256 # game loop
257 while True:
258     # event handle
259     for event in pygame.event.get():
260         # 종료
261         if event.type == QUIT or (event.type == KEYUP and event.key == K_ESCAPE):
262             pygame.quit()
263             sys.exit()
264         if event.type == KEYDOWN:
265             if event.key == K_UP:
266                 airplane.y_change = -5
267             elif event.key == K_DOWN:
268                 airplane.y_change = 5
269             if event.key == K_RIGHT:
270                 airplane.x_change = 5
271             elif event.key == K_LEFT:
272                 airplane.x_change = -5
273             if event.key == K_LCTRL:
274                 # 속도를 증가한다.
275                 bullet_group.add(AirplaneBullet(airplane.position()))
276                 sprite_group.add(bullet_group)
277         if event.type == KEYUP:
278             if event.key == K_UP or event.key == K_DOWN:
279                 airplane.y_change = 0
280             elif event.key == K_RIGHT or event.key == K_LEFT:
281                 airplane.x_change = 0
282     # 누르고 있으면 비행기가 움직이는 부분이다.
283     # for event 내에서는 키가 떼어지거나 붙어있지만 작동하기 때문에 밖에서 처리한다.
284     airplane.change_x(airplane.x_change)
285     airplane.change_y(airplane.y_change)
```

7. 적과 충돌하기

```
287 # 배경 위치 설정
288 background_x -= BACKGROUNDSPEED
289 if background_x == -BACKGROUNDWIDTH:
290     background_x = BACKGROUNDWIDTH
291 draw_object(IMAGESDICT["background"], background_x, 0)
292
293 other_background_x -= BACKGROUNDSPEED
294 if other_background_x == -BACKGROUNDWIDTH:
295     other_background_x = BACKGROUNDWIDTH
296 draw_object(IMAGESDICT["background"], other_background_x, 0)
297
298 # 박쥐가 죽으면 재시작 시간 이후 박쥐를 만든다.
299 if BatEnemy.BATTIME <= time.time() - BatEnemy.bat_remove_time \
300     and BatEnemy.bat_num <= 0:
301     bat_group.add(BatEnemy())
302     sprite_group.add(bat_group)
303
304 # fireball 생성
305 if len(fireball_group) <= 0:
306     fireball_group.add(FireBall())
307     sprite_group.add(fireball_group)
```


7. 적과 충돌하기

```
331 # 충돌을 검사한다.
332 # 박쥐와 총알의 충돌 검사
333 bat_collision_dict = pygame.sprite.groupcollide(bullet_group, bat_group, False, False)
334 if bat_collision_dict:
335     for bullet in bat_collision_dict.keys():
336         bat_x, bat_y = bat_collision_dict[bullet][0].position()
337         boom_group.add(Boom(bat_x, bat_y))
338         sprite_group.add(boom_group)
339         pygame.sprite.groupcollide(bullet_group, bat_group, True, True)
340 # 비행기와 박쥐, 파이어볼의 충돌 검사
341 airplane_crash_bat = pygame.sprite.spritecollide(airplane, bat_group, False)
342 airplane_crash_fire = pygame.sprite.spritecollide(airplane, fireball_group, False)
343 if airplane_crash_bat or airplane_crash_fire:
344     # group이 그냥 초기화되면, 소멸자가 작동하지 않는 것으로 보아, 객체가 남는 것으로 보인다.
345     # group을 명시적으로 비워준다.
346     if airplane_crash_bat:
347         # 박쥐의 비활성 충돌시, 리스트로 batsprite를 넘겨줘서 bat객체가 삭제되지 않는 것으로 추정. 비워준다.
348         del airplane_crash_bat[:]
349     sprite_group.empty()
350     bullet_group.empty()
351     bat_group.empty()
352     boom_group.empty()
353     fireball_group.empty()
354     crash()
355
356 # 모든 sprite의 update함수를 실행한다.
357 sprite_group.update()
358
359 # 다른 스프라이트 그리기
360 sprite_group.draw(DISPLAYSURF)
361
362 pygame.display.update()
363 FPSLOCK.tick(FPS)
```

7. 적과 충돌하기

```
341 def game_init():
342     global FPSLOCK, DISPLAYSURF
343     global IMAGESDICT
344     FPSLOCK = pygame.time.Clock()
345     pygame.init()
346
347     # DISPLAY Surface 설정하기
348     DISPLAYSURF = pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT))
349     pygame.display.set_caption('PyFlying')
350
351     # 이미지 받아오기
352     IMAGESDICT = {"airplane": pygame.image.load('images/plane.png'),
353                  "background": pygame.image.load('images/background.png'),
354                  "bat": pygame.image.load('images/bat.png'),
355                  "fireball1": pygame.image.load('images/fireball.png'),
356                  "fireball2": pygame.image.load('images/fireball2.png'),
357                  "bullet": pygame.image.load('images/bullet.png'),
358                  "boom": pygame.image.load('images/boom.png'),
359                  "blank": pygame.image.load('images/blank.png')}
360
361     # 배경 이미지 게임 윈도우 크기에 맞추기
362     assert WINDOWWIDTH <= ORIGINBACKGROUNDWIDTH or WINDOWHEIGHT <= ORIGINBACKGROUNDHEIGHT, \
363            '게임 윈도우 크기가 너무 큼니다.'
364     IMAGESDICT["background"] = pygame.transform.scale(IMAGESDICT["background"], (WINDOWWIDTH, WINDOWHEIGHT))
365     main()
366
367 if __name__ == '__main__':
368     game_init()
369
370
```

7. 적과 충돌하기

```
26 class AirPlane(pygame.sprite.Sprite):
27     """
31     def __init__(self):
32         global IMAGESDICT
33         super().__init__()
34         self.image = IMAGESDICT['airplane']
35         self.rect = self.image.get_rect()
36         self.rect.left = WINDOWWIDTH * 0.05
37         self.rect.top = WINDOWWIDTH * 0.8
```

- AirPlane class이다.
- player이기 때문에 위치를 정해줄 필요가 없다.
- 외부에 있었던 모든 airplane관련된 변수와 함수를 집어 넣었다.

```

39 def change_y(self, value):
40     if self.rect.top + value < 0:
41         self.rect.top = 0
42     elif self.rect.top + value > WINDOWHEIGHT - self.image.get_height():
43         self.rect.top = WINDOWHEIGHT - self.image.get_height()
44     else:
45         self.rect.top += value
46
47 def change_x(self, value):
48     if self.rect.left + value < 0:
49         self.rect.left = 0
50     elif self.rect.left + value > WINDOWWIDTH - self.image.get_width():
51         self.rect.left = WINDOWWIDTH - self.image.get_width()
52     else:
53         self.rect.left += value
54
55 def position(self):
56     return self.rect.left, self.rect.top

```

- 비행기의 x, y의 변화를 여기서 처리한다.
- 이때, 화면 밖으로 넘어가지 못하게 막는다.
- position은 총알을 만들기 위해 위치를 되돌려주는 부분이다.

7. 적과 충돌하기

- instance 변수인 rect를 외부에서 접근해서 작업할 수도 있지만, 접근하지 않고 메서드를 만들어서 작업하였다.
- 이는 객체지향의 특성인 캡슐화 및 은닉화를 지키기 위해서 만들었다.

7. 적과 충돌하기

- 객체지향의 특성
 - 추상화: 공통의 속성이나 기능을 묶어 이름을 붙인다.
ex) 개, 고양이, 말 등을 네발 동물이라는 객체로 묶어 부른다.
 - 은닉화: 객체 내부의 변수값을 외부에서 수정하거나 조작하지 못하게 만드는 것. 데이터의 수정은 메서드를 통해서만 접근한다.
ex) 은닉화가 안되면..? -npc 돈 빼가기 등이 가능하다.
 - 캡슐화: 데이터 구조와 데이터를 다루는 방법들을 묶는 것객체가 맡은 역할을 수행하기 위한 하나의 목적을 묶는다.
ex) 캡슐로 된 약을 먹으면 내부 캡슐이 위, 십이지장, 소장 등에서 각각 녹아 흡수된다.
캡슐화가 되면 은닉화도 효력이 나타난다.

7. 적과 충돌하기

- 객체지향의 특성
 - 상속성, 재사용(Inheritance): 상위 객체의 특징을 하위 개념이 물려받는 것
ex) 자동차라는 부모클래스-> 지붕 뚜껑이 열리게 만든다.
 - 다형성(Polymorphiism): 부모클래스에서 물려받은 함수를 자식클래스에서 오버라이딩 해서 사용하는 것
ex) 스포츠카, 트럭, 승용차 등의 객체를 만들었다. 차가 다르면 각각 다른 함수로 움직이게 만들지 않고, 어떤 차든 움직이게 만드는 것.
sprite.group의 update를 생각한다.

<http://88240.tistory.com/228>

<http://showmiso.tistory.com/116>

7. 적과 충돌하기

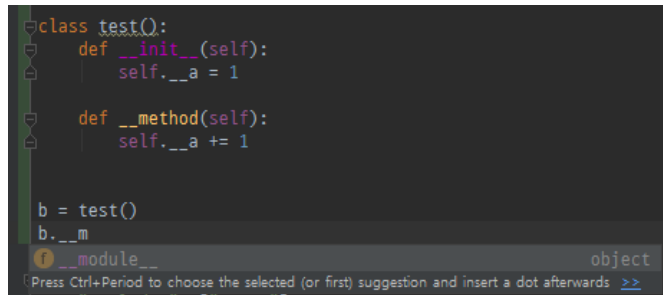
- 보통 다른 언어는 변수를 외부에서 접근할 수 있는지를 public/private/protected라는 것을 두어 외부에서 접근을 제한하기도 한다.
- 하지만 이건 실제로 변경할 수는 있다.
- 파이썬에서는 프로그래머가 책임감을 갖고 보안을 중시하도록 만들었기 때문에 접근이 가능하다.
- 만일 private 변수를 모방하고 싶다면 `__`를 붙여 모방한다. (pycharm에서 사용하면, 외부에서 변수이나 메서드가 안 보인다.)

<http://88240.tistory.com/228>

<http://showmiso.tistory.com/116>

7. 적과 충돌하기

- 만일 private 변수를 모방하고 싶다면 __를 붙여 모방한다. (pycharm에서 사용하면, 외부에서 변수이나 메서드가 안 보인다.)



```
class test():
    def __init__(self):
        self.__a = 1

    def __method(self):
        self.__a += 1

b = test()
b.__m
```

Completion list: module, object

<http://pythonstudy.xyz/python/article/19->

[%ED%81%B4%EB%9E%98%EC%8A%A4](http://pythonstudy.xyz/python/article/19-%ED%81%B4%EB%9E%98%EC%8A%A4)

<http://hashcode.co.kr/questions/2301/%ED%8C%8C%EC%9D%B4%EC%8D%AC%EC%9D%98->

[%ED%81%B4%EB%9E%98%EC%8A%A4%EC%97%90%EB%8F%84-private-](http://hashcode.co.kr/questions/2301/%ED%8C%8C%EC%9D%B4%EC%8D%AC%EC%9D%98-%ED%81%B4%EB%9E%98%EC%8A%A4%EC%97%90%EB%8F%84-private-)

[%EB%B3%80%EC%88%98%EA%B0%80-](http://hashcode.co.kr/questions/2301/%ED%8C%8C%EC%9D%B4%EC%8D%AC%EC%9D%98-%ED%81%B4%EB%9E%98%EC%8A%A4%EC%97%90%EB%8F%84-private-%EB%B3%80%EC%88%98%EA%B0%80-)

[%EC%9E%88%EB%82%98%EC%9A%94](http://hashcode.co.kr/questions/2301/%ED%8C%8C%EC%9D%B4%EC%8D%AC%EC%9D%98-%ED%81%B4%EB%9E%98%EC%8A%A4%EC%97%90%EB%8F%84-private-%EB%B3%80%EC%88%98%EA%B0%80-%EC%9E%88%EB%82%98%EC%9A%94)

7. 적과 충돌하기

- `_bar`와 같이 쓸 경우 접근이 막혀 있지 않아도 그러지 말라는 의미가 된다.

7. 적과 충돌하기

- fireball: 앞에서 만든 것을 class화 시킴

```
class FireBall(pygame.sprite.Sprite):  
    FIREBALLSPEED = 15  
    FIREBALLNUM = 1  
    PROBABILITY = 7  
  
    def __init__(self):  
        global IMAGESDICT  
        super().__init__()   
        self.fireball_choice = random.randint(1, self.PROBABILITY)  
        if self.fireball_choice <= 2:  
            self.image = IMAGESDICT["fireball%s" % self.fireball_choice]  
            self.rect = self.image.get_rect()  
            self.rect.left, self.rect.top = init_enemy_pos(self.image)  
        else:  
            self.image = IMAGESDICT["blank"]  
            self.rect = self.image.get_rect()  
            self.rect.left = WINDOWWIDTH  
            self.rect.top = -2
```

7. 적과 충돌하기

- fireball: 앞에서 만든 것을 class화 시킴

```
def update(self):  
    if self.rect.left <= 0:  
        self.kill()  
    if self.fireball_choice <= 2:  
        self.rect.left -= self.FIREBALLSPEED  
    else:  
        self.rect.left -= 2 * self.FIREBALLSPEED
```

7. 적과 충돌하기

- text surface와 rect를 되돌려준다.

```
def text_obj(text, font, color):  
    text_surface = font.render(text, True, color)  
    return text_surface, text_surface.get_rect()
```

7. 적과 충돌하기

```
def disp_message(sentence, pos_x, pos_y, size, color):  
    text = pygame.font.Font('freesansbold.ttf', int(size*WINDOWWIDTH/ORIGINBACKWIDTH))  
    text_surf, text_rect = text_obj(sentence, text, color)  
    text_rect.center = (pos_x, pos_y)  
    draw_object(text_surf, text_rect.left, text_rect.top)
```

- 메시지를 표시하는 함수로 text, x위치, y위치, size, color를 받아 위치에 text를 만든다.
- rect.center를 통해 중앙에 배치한다.

7. 적과 충돌하기

- 적과 충돌하면, 충돌이라는 글을 정중앙에 크게 만든다. 그리고 2초 동안 표시한 이후 초기화 한다.

```
def crash():  
    disp_message("Crashed!", WINDOWWIDTH/2, WINDOWHEIGHT/2, 115, RED)  
    pygame.display.update()  
    pygame.time.delay(2000)  
    main()
```

7. 적과 충돌하기

- airplane의 위치를 받아와서 총알의 생성에 넘겨준다.

```
bullet_group.add(AirplaneBullet(airplane.position()))  
sprite_group.add(bullet_group)
```

- 변화 값을 넘겨줘 airplane내부에서 처리하도록 한다. 은닉화의 예시

```
airplane.change_x(airplane_x_change)  
airplane.change_y(airplane_y_change)
```


7. 적과 충돌하기

- fireball_group내에 fireball이 없으면 생성한다.

```
# fireball 생성
if len(fireball_group) <= 0:
    fireball_group.add(FireBall())
    sprite_group.add(fireball_group)
```

7. 적과 충돌하기

- airplane sprite와 bat_group, fireball_group간의 충돌을 확인한다.

```
airplane_crash_bat = pygame.sprite.spritecollide(airplane, bat_group, False)
airplane_crash_fire = pygame.sprite.spritecollide(airplane, fireball_group, False)
if airplane_crash_bat or airplane_crash_fire:
```

- group collide와는 다르게, True를 넣으면 group내에 있는 sprite만 사라진다.
- 리스트내에 sprite를 넣어 되돌려준다.

7. 적과 충돌하기

```
if airplane_crash_bat:  
    # 박쥐와 비행기 충돌시, 리스트  
    del airplane_crash_bat[:]
```

- bat이 리스트에 들어있을 경우 소멸자가 작동하지 않아 명시적으로 비워주었다.
- del list는 list 자체를 소멸시킨다.
- del list[:]는 list를 남겨두고 내부 내용만 소멸시킨다.

<https://dongyeopblog.wordpress.com/2016/02/21/python-%EB%A6%AC%EC%8A%A4%ED%8A%B8-%EC%B4%88%EA%B8%B0%ED%99%94/>

7. 적과 충돌하기

- 명시적으로 group을 비운다.
- 그리고 crash()를 호출하여 충돌했다는 표시를 띄운다.
- 모두 sprite_group에 묶여있어서 .update()를 통해 전부 업데이트가 가능하다.

```
sprite_group.empty()
bullet_group.empty()
bat_group.empty()
boom_group.empty()
fireball_group.empty()
crash()

# 모든 sprite의 update함수를 실행한다
sprite_group.update()
```

8. 새로운 룰 만들기

- 잡은/넘어간 박쥐 개수 띄우기
- 박쥐 잡은 개수가 늘어나면 박쥐와 파이어볼 개수 늘리기
- 넘어간 박쥐가 4마리 이상 넘어가면 죽게 만들기

8. 새로운 룰

```
import pygame
import sys
import random
import time
from pygame.locals import *

# 상수영역
# 초당 프레임 수
FPS = 30
# 윈도우 크기, 비율 일정하게 만들
WINDOWWIDTH = 1080
WINDOWHEIGHT = int(WINDOWWIDTH / 2)
# 배경 최대 크기
ORIGINBACKGROUNDWIDTH = 1280
ORIGINBACKGROUNDHEIGHT = 640
# 배경 속도
BACKGROUNDSPPEED = 2
# 색
WHITE = (255, 255, 255)
RED = (255, 0, 0)
```

```
class AirPlane(pygame.sprite.Sprite):
    """
    """
    bat_max_catch = 0

    def __init__(self):
        global IMAGESDICT
        super().__init__()
        self.image = IMAGESDICT['airplane']
        self.rect = self.image.get_rect()
        self.rect.left = WINDOWWIDTH * 0.05
        self.rect.top = WINDOWHEIGHT * 0.8
        self.bat_catch = 0

    def change_y(self, value):
        """
        """
        if self.rect.top + value < 0:
            self.rect.top = 0
        elif self.rect.top + value > WINDOWHEIGHT - self.image.get_height():
            self.rect.top = WINDOWHEIGHT - self.image.get_height()
        else:
            self.rect.top += value

    def change_x(self, value):
        """
        """
        if self.rect.left + value < 0:
            self.rect.left = 0
        elif self.rect.left + value > WINDOWWIDTH - self.image.get_width():
            self.rect.left = WINDOWWIDTH - self.image.get_width()
        else:
            self.rect.left += value

    def position(self):
        """
        """
        return self.rect.left, self.rect.top

    def bat_catch_add(self):
        self.bat_catch += 1

    def bat_catch_return(self):
        return self.bat_catch
```

8. 새로운 룰 만들기

```
class Fireball(pygame.sprite.Sprite):
    """Fireball"""
    FIREBALLSPEED = 15
    FIREBALLNUM = 1
    PROBABILITY = 7

    def __init__(self):
        global IMAGESDICT
        super().__init__()
        self.fireball_choice = random.randint(1, self.PROBABILITY)
        if self.fireball_choice <= 2:
            self.image = IMAGESDICT["fireballs" % self.fireball_choice]
            self.rect = self.image.get_rect()
            self.rect.left, self.rect.top = init_enemy_pos(self.image)
        else:
            self.image = IMAGESDICT["blank"]
            self.rect = self.image.get_rect()
            self.rect.left = WINDOWWIDTH
            self.rect.top = -2

    def update(self):
        if self.rect.left <= 0:
            self.kill()
        if self.fireball_choice <= 2:
            self.rect.left -= self.FIREBALLSPEED
        else:
            self.rect.left -= 2 * self.FIREBALLSPEED

class Boom(pygame.sprite.Sprite):
    """Boom"""
    BOOMTIME = 5

    def __init__(self, x, y):
        global IMAGESDICT
        super().__init__()
        self.image = IMAGESDICT["boom"]
        self.rect = self.image.get_rect()
        self.rect.left = x
        self.rect.top = y
        self.time = 0

    def update(self):
        self.time += 1
        if self.time >= self.BOOMTIME:
            self.kill()
```

```

138 class BatEnemy(pygame.sprite.Sprite):
139     """
140     """
141     BATSPEED = 7
142     BATTIME = 3
143     bat_num = 0
144     bat_remove_time = []
145     bat_passed = 0
146
147     def __init__(self):
148         """
149         """
150         global IMAGESDICT
151         super().__init__()
152         self.image = IMAGESDICT["bat"]
153         self.rect = self.image.get_rect()
154         self.rect.left, self.rect.top = init_enemy_pos(self.image)
155         BatEnemy.bat_num += 1
156
157     def __del__(self):
158         """
159         """
160         BatEnemy.bat_num -= 1
161         BatEnemy.bat_remove_time.append(time.time())
162         if len(BatEnemy.bat_remove_time) >= 2 and \
163             BatEnemy.bat_remove_time[-2] - BatEnemy.bat_remove_time[-1] < 0.1:
164             BatEnemy.bat_remove_time[-1] += 0.5
165
166     def update(self):
167         """
168         """
169         self.rect = self.rect.move(-self.BATSPEED, 0)
170         if self.rect.left < 0:
171             BatEnemy.bat_passed += 1
172             self.kill()
173
174     def position(self):
175         """
176         """
177         return self.rect.left, self.rect.top
178
179
180 class AirplaneBullet(pygame.sprite.Sprite):
181     """
182     """
183     BULLETSPEED = 15
184
185     def __init__(self, airplane_xy):
186         """
187         """
188         global IMAGESDICT
189         super().__init__()
190         self.image = IMAGESDICT["bullet"]
191         self.rect = self.image.get_rect()
192         self.rect.left = airplane_xy[0] + IMAGESDICT["airplane"].get_width()
193         self.rect.top = airplane_xy[1] + IMAGESDICT["airplane"].get_height() / 2
194
195     def update(self):
196         """
197         """
198         self.rect = self.rect.move(self.BULLETSPEED, 0)
199         if self.rect.left > WINDOWWIDTH:
200             self.kill()

```


8. 새로운 룰 만들기

```
220 def text_obj(text, font, color):
221     문자 객체 생성
222     text_surface = font.render(text, True, color)
223     return text_surface, text_surface.get_rect()
224
225
226 def disp_message(sentence, pos_x, pos_y, size, color, position=""):
227     메시지 표시
228     text = pygame.font.Font('freesansbold.ttf', int(size*WINDOWWIDTH/ORIGINBACKGROUNDWIDTH))
229     text_surf, text_rect = text_obj(sentence, text, color)
230     if position == "center":
231         text_rect.center = (pos_x, pos_y)
232     else:
233         text_rect.left = pos_x
234         text_rect.top = pos_y
235     draw_object(text_surf, text_rect.left, text_rect.top)
236
237
238 def game_over(bat_captured, text):
239     게임 종료
240     disp_message(text, WINDOWWIDTH/2, WINDOWHEIGHT/2, 115, RED, "center")
241     BatEnemy.bat_passed = 0
242     # 가지고 있는 재생성 시간을 다 초기화 한다.
243     del(BatEnemy.bat_remove_time[:])
244     # 가끔 초기화 되지 않는 오류가 있어 명시적으로 초기화 한다.
245     BatEnemy.bat_num = 0
246     AirPlane.bat_max_catch = max(bat_captured, AirPlane.bat_max_catch)
247     pygame.display.update()
248     pygame.time.delay(2000)
249     main()
```

crash가 지워짐

8. 새로운 룰 만들기

```
272 def draw_bat_score(bat_captured):
273     """ """
274     disp_message("Top Score: %d, Bat captured: %d, Bat passed: %d"
275                 % (AirPlane.bat_max_catch, bat_captured, BatEnemy.bat_passed),
276                 5, 5, 20, WHITE)
277
278
279 def recreate_bat():
280     """ """
281     if BatEnemy.BATTIME <= time.time() - BatEnemy.bat_remove_time[0]:
282         BatEnemy.bat_remove_time.pop(0)
283         return True
284     else:
285         return False
286
287
288 def init_enemy_pos(image):
289     """ """
290     x = WINDOWWIDTH
291     y = random.randrange(0, WINDOWHEIGHT - image.get_height())
292     return x, y
293
294
295 def draw_object(image, x, y):
296     """ """
297     global DISPLAYSURF
298     DISPLAYSURF.blit(image, (x, y))
```

8. 새로운

```
320 def main():
321     global FPSLOCK, DISPLAYSURF
322     global IMAGESDICT
323
324     # 비행기 x, y 변화값을 0으로 초기화 한다.
325     airplane_y_change = 0
326     airplane_x_change = 0
327
328     # 윈도우 변경에 따른 배경크기 변경
329     BACKGROUNDWIDTH = IMAGESDICT["background"].get_width()
330
331     # 배경 초기 위치
332     background_x = 0
333     other_background_x = BACKGROUNDWIDTH
334
335     # 개별 sprite group
336     bullet_group = pygame.sprite.Group()
337     bat_group = pygame.sprite.Group()
338     boom_group = pygame.sprite.Group()
339     fireball_group = pygame.sprite.Group()
340     # 전체 sprite group
341     sprite_group = pygame.sprite.Group()
342
343     # 비행기 sprite
344     airplane = AirPlane()
345     sprite_group.add(airplane)
346
347     # 박쥐 및 fireball 최대 개수 초기화
348     bat_maximum_num = 1
349     fireball_max_num = 1
350
351     # game loop
352     while True:
353         # event handle
354         for event in pygame.event.get():
355             # 종료
356             if event.type == QUIT or (event.type == KEYUP and event.key == K_ESCAPE):
357                 pygame.quit()
358                 sys.exit()
359
360             if event.type == KEYDOWN:
361                 if event.key == K_UP:
362                     airplane_y_change = -5
363                 elif event.key == K_DOWN:
364                     airplane_y_change = 5
365                 if event.key == K_RIGHT:
366                     airplane_x_change = 5
367                 elif event.key == K_LEFT:
368                     airplane_x_change = -5
369                 if event.key == K_LCTRL:
370                     # 총알을 추가한다.
371                     bullet_group.add(AirplaneBullet(airplane.position()))
372                     sprite_group.add(bullet_group)
373
374             if event.type == KEYUP:
375                 if event.key == K_UP or event.key == K_DOWN:
376                     airplane_y_change = 0
377                 elif event.key == K_RIGHT or event.key == K_LEFT:
378                     airplane_x_change = 0
379
380             # 그리고 있으면 비행기가 움직이는 부분이다.
381             if event.type == KEYUP:
382                 if event.key == K_UP or event.key == K_DOWN:
383                     airplane_y_change = 0
384                 elif event.key == K_RIGHT or event.key == K_LEFT:
385                     airplane_x_change = 0
386
387             # 배경 위치 설정
388             background_x -= BACKGROUNDSPD
389             if background_x < -BACKGROUNDWIDTH:
390                 background_x = BACKGROUNDWIDTH
391             other_background_x -= BACKGROUNDSPD
392             if other_background_x < -BACKGROUNDWIDTH:
393                 other_background_x = BACKGROUNDWIDTH
394
395             # 배경 위치 설정
396             background_x -= BACKGROUNDSPD
397             if background_x < -BACKGROUNDWIDTH:
398                 background_x = BACKGROUNDWIDTH
399             other_background_x -= BACKGROUNDSPD
400             if other_background_x < -BACKGROUNDWIDTH:
401                 other_background_x = BACKGROUNDWIDTH
```

8. 새로운 룰 만들기

```
392 # 박쥐가 죽으면 재시작 시간 이후 박쥐를 만든다.
393 if bat_maximum_num > BatEnemy.bat_num:
394     if BatEnemy.bat_remove_time:
395         if recreate_bat():
396             bat_group.add(BatEnemy())
397             sprite_group.add(bat_group)
398     else:
399         bat_group.add(BatEnemy())
400         sprite_group.add(bat_group)
401
402 # fireball 생성
403 if len(fireball_group) < fireball_max_num:
404     fireball_group.add(FireBall())
405     sprite_group.add(fireball_group)
406
407 # 충돌을 검사한다.
408 # 박쥐와 총알의 충돌 검사
409 bat_collision_dict = pygame.sprite.groupcollide(bullet_group, bat_group, False, False)
410 if bat_collision_dict:
411     for bullet in bat_collision_dict.keys():
412         bat_x, bat_y = bat_collision_dict[bullet][0].position()
413         boom_group.add(Boom(bat_x, bat_y))
414         sprite_group.add(boom_group)
415         airplane.bat_catch_add()
416         pygame.sprite.groupcollide(bullet_group, bat_group, True, True)
417 # 박쥐, fireball의 숫자를 규칙에 따라 늘린다.
418 if airplane.bat_catch_return() % 2 == 0:
419     bat_maximum_num += 1
420     BatEnemy.bat_remove_time.insert(0, time.time())
421 if airplane.bat_catch_return() % 4 == 0:
422     fireball_max_num += 1
```

8. 새로운 룰 만들기

```
424 # 비행기와 박쥐, 파이어볼의 충돌 검사
425 airplane_crash_bat = pygame.sprite.spritecollide(airplane, bat_group, True)
426 airplane_crash_fire = pygame.sprite.spritecollide(airplane, fireball_group, True)
427 if airplane_crash_bat or airplane_crash_fire:
428     # group이 그냥 초기화되면, 소멸자가 작동하지 않는 것으로 보아, 객체가 남은 것으로 보인다.
429     # group을 명시적으로 비워준다.
430     if airplane_crash_bat:
431         # 박쥐와 비행기 충돌시, 리스트로 batsprite를 넘겨줘서 bat객체와 삭제되지 않는 것으로 추
432         del airplane_crash_bat[:]
433     sprite_group.empty()
434     bullet_group.empty()
435     bat_group.empty()
436     boom_group.empty()
437     fireball_group.empty()
438     game_over(airplane.bat_catch_return(), "Crashed!")
439
440 # 박쥐를 잡은 개수와 넘어간 개수를 띄운다.
441 draw_bat_score(airplane.bat_catch_return())
442
443 # 박쥐가 4마리 이상 넘어가면 gameover를 출력한다.
444 if BatEnemy.bat_passed >= 4:
445     game_over(airplane.bat_catch_return(), "Game Over")
446
447 # 모든 sprite의 update함수를 실행한다.
448 sprite_group.update()
449
450 # 다른 스프라이트 그리기
451 sprite_group.draw(DISPLAYSURF)
452
453 pygame.display.update()
454 FPSLOCK.tick(FPS)
```

8. 새로운 룰 만들기

```
457 def game_init():
458     """
459     global FPSLOCK, DISPLAYSURF
460     global IMAGESDICT
461     FPSLOCK = pygame.time.Clock()
462     pygame.init()
463
464     # DISPLAY Surface 설정하기
465     DISPLAYSURF = pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT))
466     pygame.display.set_caption('PyFlying')
467
468     # 이미지 받아오기
469     IMAGESDICT = {'airplane': pygame.image.load('images/plane.png'),
470                  'background': pygame.image.load('images/background.png'),
471                  'bat': pygame.image.load('images/bat.png'),
472                  'fireball1': pygame.image.load('images/fireball.png'),
473                  'fireball2': pygame.image.load('images/fireball2.png'),
474                  'bullet': pygame.image.load('images/bullet.png'),
475                  'boom': pygame.image.load('images/boom.png'),
476                  'blank': pygame.image.load('images/blank.png')}
477
478     # 배경 이미지 게임 윈도우 크기에 맞추기
479     assert WINDOWWIDTH <= ORIGINBACKGROUNDWIDTH or WINDOWHEIGHT <= ORIGINBACKGROUNDHEIGHT, \
480            "게임 윈도우 크기가 너무 큼니다."
481     IMAGESDICT['background'] = pygame.transform.scale(IMAGESDICT['background'], (WINDOWWIDTH, WINDOWHEIGHT))
482     main()
483
484 if __name__ == '__main__':
485     game_init()
```

8. 새로운 룰 만들기

- 박쥐를 잡은 것은 비행기의 점수임으로, bat_catch라는 인스턴스 변수를 만들었다..
- bat_max_catch는 게임이 한 번 끝나도 계속 가지고 있어야 함으로 class변수로 만들었다.

```
class AirPlane(pygame.sprite.Sprite):  
    bat_max_catch = 0  
  
    def __init__(self):  
        global IMAGESDICT  
        super().__init__()   
        self.image = IMAGESDICT['airplane']  
        self.rect = self.image.get_rect()  
        self.rect.left = WINDOWWIDTH * 0.05  
        self.rect.top = WINDOWWIDTH * 0.8  
        self.bat_catch = 0
```

8. 새로운 룰 만들기

- 총알에 의해 박쥐가 죽으면, bat_catch를 늘리는 메서드를 만들었다.
- 또한, 점수를 출력해야 되기 때문에 점수를 되돌려주는 메서드를 만들었다.

```
def bat_catch_add(self):  
    self.bat_catch += 1  
  
def bat_catch_return(self):  
    return self.bat_catch
```


8. 새로운 룰 만들기

- 박쥐가 여러 마리가 되어서 `remove_time`을 리스트로 변경하였다.
- 또한, 박쥐가 몇 마리 통과했는지 class변수로 나타내었다.

```
class BatEnemy(pygame.sprite.Sprite):  
    BATSPEED = 7  
    BATTIME = 3  
    bat_num = 0  
    bat_remove_time = []  
    bat_passed = 0
```

8. 새로운 룰 만들기

```
def __del__(self):  
    BatEnemy.bat_num -= 1  
    BatEnemy.bat_remove_time.append(time.time())  
    if len(BatEnemy.bat_remove_time) >= 2 and \  
        BatEnemy.bat_remove_time[-2]-BatEnemy.bat_remove_time[-1] < 0.1:  
        BatEnemy.bat_remove_time[-1] += 0.5
```

- 박쥐가 죽으면, remove_time에 시간을 입력한다.
- 또 박쥐가 두 마리 이상 죽었을 때, 시간이 0.1초 이내라면 0.5초를 더해 나오는 시간을 조절한다.
 - 너무 붙어있는 경우가 있어 시간을 추가하는 것이다.

8. 새로운 룰 만들기

```
def disp_message(sentence, pos_x, pos_y, size, color, position=""):
    text = pygame.font.Font('freesansbold.ttf', int(size*WINDOWWIDTH/ORIGINBACKGROUNDWIDTH))
    text_surf, text_rect = text_obj(sentence, text, color)
    if position == "center":
        text_rect.center = (pos_x, pos_y)
    else:
        text_rect.left = pos_x
        text_rect.top = pos_y
    draw_object(text_surf, text_rect.left, text_rect.top)
```

- disp_message를 재사용 가능하도록 수정하였다.
- position에 center라는 값이 입력되면 중앙에 pos_x, pos_y를 넣도록 만들고, 아닐 경우 일반적인 방식인 왼쪽 위를 기준으로 만들도록 하였다.

8. 새로운 룰 만들기

- crash와 gameover를 각각 만들지 않고, 통합해서 재사용하도록 만들었다.
- 메시지를 받아서 출력한다.
- 재시작전에 초기화 해야되는 값들을 초기화 한다.

```
def game_over(bat_captured, text):  
    disp_message(text, WINDOWWIDTH/2, WINDOWHEIGHT/2, 115, RED, "center")  
    BatEnemy.bat_passed = 0  
    # 가지고 있는 재생성 시간을 다 초기화 한다.  
    del(BatEnemy.bat_remove_time[:])  
    # 가끔 초기화 되지 않는 오류가 있어 명시적으로 초기화 한다.  
    BatEnemy.bat_num = 0  
    AirPlane.bat_max_catch = max(bat_captured, AirPlane.bat_max_catch)  
    pygame.display.update()  
    pygame.time.delay(2000)  
    main()
```

8. 새로운 룰 만들기

- bat_passed값, bat_remove_time값
- bat_num은 어디선가 버그로 인해 초기화 안되는 경우가 있어 강제로 초기화 한다.
- 최고 스코어를 저장하고 시간이 지난 후 초기화 한다.

```
def game_over(bat_captured, text):  
    disp_message(text, WINDOWWIDTH/2, WINDOWHEIGHT/2, 115, RED, "center")  
    BatEnemy.bat_passed = 0  
    # 가지고 있는 재생성 시간을 다 초기화 한다.  
    del(BatEnemy.bat_remove_time[:])  
    # 가끔 초기화 되지 않는 오류가 있어 명시적으로 초기화 한다.  
    BatEnemy.bat_num = 0  
    AirPlane.bat_max_catch = max(bat_captured, AirPlane.bat_max_catch)  
    pygame.display.update()  
    pygame.time.delay(2000)  
    main()
```

8. 새로운 룰 만들기

- score을 표시하는 부분이다.
- 값을 받아서 disp_message를 통해 표시한다.

```
def draw_bat_score(bat_captured):  
    disp_message("Top Score: %d, Bat captured: %d, Bat passed: %d"  
                 % (AirPlane.bat_max_catch, bat_captured, BatEnemy.bat_passed),  
                 5, 5, 20, WHITE)
```

8. 새로운 룰 만들기

- bat을 재생성해도 되는지 확인하는 부분이다.
- bat_remove_time 중 맨 첫번째 값과 현재 시간의 차이가 BATTIME보다 커지면 True를 반환해 생성할 수 있게 한다.
- pop을 통해 맨 첫번째를 삭제한다.
- 1/30초마다 확인하기 때문에 맨 첫번째만 확인해도 괜찮다.

```
def recreate_bat():  
    if BatEnemy.BATTIME <= time.time() - BatEnemy.bat_remove_time[0]:  
        BatEnemy.bat_remove_time.pop(0)  
        return True  
    else:  
        return False
```

8. 새로운 룰 만들기

```
# 박쥐 및 fireball 최대 개수 초기화  
bat_maximum_num = 1  
fireball_max_num = 1
```

- 제어 class를 만들지 않고, main에서 처리한다.

8. 새로운 룰 만들기

- bat_maximum_num보다 bat.num이 작으면 박쥐를 만들지 확인한다.
- remove_time이 존재하는지 확인하고, 있으면 recreate_bat함수를 불러 만들지 확인한다.
- remove_time이 없으나 bat_num이 작을 경우 만들게 한다. -나중에 혹시 추가할 때 오류가 발생할 여지를 줄인다.

```
# 박쥐가 죽으면 재시작 시간 이후 박쥐를 만든다.  
if bat_maximum_num > BatEnemy.bat_num:  
    if BatEnemy.bat_remove_time:  
        if recreate_bat():  
            bat_group.add(BatEnemy())  
            sprite_group.add(bat_group)  
    else:  
        bat_group.add(BatEnemy())  
        sprite_group.add(bat_group)
```

8. 새로운 룰 만들기

```
# fireball 생성
if len(fireball_group) < fireball_max_num:
    fireball_group.add(FireBall())
    sprite_group.add(fireball_group)
```

- fireball_max_num보다 fireball의 개수가 작으면 생성해서 넣는다.

8. 새로운 룰 만들기

- 충돌 이후에 규칙에 따라 박쥐와 fireball의 숫자를 늘리는 부분이다.
- 게임난이도와 관련되어 있으니 적당히 바꿔본다. 지금은 확 어려워짐.

```
bat_collision_dict = pygame.sprite.groupcollide(bullet_group, bat_group, False, False)
if bat_collision_dict:
    for bullet in bat_collision_dict.keys():
        bat_x, bat_y = bat_collision_dict[bullet][0].position()
        boom_group.add(Boom(bat_x, bat_y))
    sprite_group.add(boom_group)
    airplane.bat_catch_add()
    pygame.sprite.groupcollide(bullet_group, bat_group, True, True)
    # 박쥐, fireball의 숫자를 규칙에 따라 늘린다.
    if airplane.bat_catch_return() % 2 == 0:
        bat_maximum_num += 1
        BatEnemy.bat_remove_time.insert(0, time.time())
    if airplane.bat_catch_return() % 4 == 0:
        fireball_max_num += 1
```

8. 새로운 룰 만들기

- bat의 최대치를 추가한 이후에 remove_time에 현재 시간을 추가하여 만들어지게 만든다.

```
bat_collision_dict = pygame.sprite.groupcollide(bullet_group, bat_group, False, False)
if bat_collision_dict:
    for bullet in bat_collision_dict.keys():
        bat_x, bat_y = bat_collision_dict[bullet][0].position()
        boom_group.add(Boom(bat_x, bat_y))
    sprite_group.add(boom_group)
    airplane.bat_catch_add()
    pygame.sprite.groupcollide(bullet_group, bat_group, True, True)
    # 박쥐, fireball의 숫자를 규칙에 따라 늘린다.
    if airplane.bat_catch_return() % 2 == 0:
        bat_maximum_num += 1
        BatEnemy.bat_remove_time.insert(0, time.time())
    if airplane.bat_catch_return() % 4 == 0:
        fireball_max_num += 1
```

8. 새로운 룰 만들기

- 레벨 디자인: 맵을 디자인 하는 것이다. 이상한 곳 빠지지 않고 잘 가도록 만든다.
 - 나무위키에 잘 정리되어 있으니 궁금하면 읽어보자.
- 게임 난이도는 크게 게임 그 자체의 난이도와 게임 시스템적인 난이도가 존재
 - 게임 그 자체의 난이도: 레벨 디자인 문제인 경우가 많다. 적이 얼마나 나오나? 공격패턴은 어떤가? 아이템의 배치는 어떤가?
 - 게임 시스템적인 난이도: 난이도나 상태이상까지 고려한 플레이어 및 적의 이동능력과 체력 기술. 세이브가 쉬운가 그런 것들.

<https://namu.wiki/w/%EB%A0%88%EB%B2%A8%20%EB%94%94%EC%9E%90%EC%9D%B8>

<https://namu.wiki/w/%EA%B2%8C%EC%9E%84%20%EB%82%9C%EC%9D%B4%EB%8F%84>

8. 새로운 룰 만들기

- RPG 레벨디자인이나 fps 레벨디자인 같은 책들이 존재한다.

<http://www.yes24.com/24/goods/17846406>

<http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9788996957638>

- 게임 자체의 디자인과 관련된 책도 많으니 궁금하면 읽어보자.

http://dnotewiki.com/note/index.php/%EA%B2%8C%EC%9E%84_%EB%94%94%EC%9E%90%EC%9D%B8/%EC%B0%B8%EA%B3%A0%EC%84%9C%EC%A0%81

8. 새로운 룰 만들기

- crash대신 game_over라는 함수를 새로 정의해서 내부의 텍스트만 다르게 만들었다.
- 이렇게 재사용성을 고려하는 것도 중요하다.

```
game_over(airplane.bat_catch_return(), "Crashed!")

# 박쥐를 잡은 개수와 넘어간 개수를 띄운다.
draw_bat_score(airplane.bat_catch_return())

# 박쥐가 4마리 이상 넘어가면 gameOver를 출력한다.
if BatEnemy.bat_passed >= 4:
    game_over(airplane.bat_catch_return(), "Game Over")
```