**총알피하기**

정용철(20203109)

조선대학교 전자공학(지능 Iot학과)

**영어 글 제목**

dodge-the-bullets

Department of Electronics Engineering, Chosun University

e-mail:jyc6863972@naver .com

**요       약**

총알을 피하는 게임에서 총알갯수 추가 속도 증가 ,게임플레이 타임 추가 다른 학점도 넣어서 먹으면 스코어 추가 스코어 기준점에 따라 엔딩화면이 달라짐

**1. 서론**

<https://github.com/toallknown/dodge-the-bullets> 다운로드 받은 프로젝트입니다

저는 총알 피하기를 총알을 먹을수 있을까? 라는 호기심에서 시작되어

지금 제가 다니고 있는 학과 기준으로 졸업을 만들자 라는 생각이 들었습니다

매우 어려운 140학점을 졸업하는 입장으로써ㅗ

스코어 제도를 만들고 추가하자고 생각했습니다

추가하다 보니 엔딩 화면도 없어서 엔딩화면으로 전환되는 게임을 만들고 싶었고

**2. 실행 방법**

* 총알이 디파인 된 것을 (A,B,F)로 늘리고 한장면에 여러 개 나올수 있게 배열로 선언하였습니다
* 총알이 삭제되는 조건을 바꾸어 학생이 총알들을 먹으면 사라지게 하고 맨아래로 떨어져도 삭제되게 바꿨습니다
* 스코어랑 시간을 다파인하여 게임시작 기준으로 0점 0초로 실행되고 각각 왼쪽위 오른쪽위로 보이게 선언 했습니다
* 스코어가 140이 넘으면 졸업 엔딩 0점미만으로 되면 학사경고로 바꾸는 엔딩화면 을 추가

했습니다

* 스코어는 기본적으로 10초가 지날때마다 1점씩추가되는 소소한 구문도 있습니다
* 시간이.지날수록 속도 추가 개수 추가 하는 선언을 만들었습니다

**3. 코드 설명 \*폴더 구성 포함**

import random

import pygame

from datetime import datetime, timedelta

임포트는 pygame ,random, datatime,timedelta

기본적으로 있던 파이게임과 총알이 랜덤위치에서 떨어지게 만드는 랜덤 ,시간을 표시하기위해 datatime,timedelta 선언하였습니다

**기본 define 구문**

|  |  |
| --- | --- |
| 원본 | 내가 만든 프로젝트 |
| pygame.init() #초기화임  #화면의 크기  screen\_width = 480 #가로 크기  screen\_height = 640 #세로 크기  screen = pygame.display.set\_mode((screen\_width, screen\_height)) #이게 화면 크기다 하하하  #FPS  clock = pygame.time.Clock()  #배경 이미지  background = pygame.image.load(r'C:\Users\chosun\Desktop\dodge-the-bullets-원본\background.png')  #솔져 스프라이트  sol = pygame.image.load(r"C:\Users\chosun\Desktop\dodge-the-bullets-원본\sol.png")  sol\_size = sol.get\_rect().size  sol\_width = sol\_size[0]  sol\_height = sol\_size[0]  sol\_x\_pos = (screen\_width / 2) - (sol\_width / 2)  sol\_y\_pos = screen\_height - sol\_height  #솔의 속도  to\_x = 0  sol\_speed = 10  ##########################################3  #총알? 미사일? 대포? 난몰라  bullet = pygame.image.load(r"C:\Users\chosun\Desktop\dodge-the-bullets-원본\bullet.png")  bullet\_size = bullet.get\_rect().size  bullet\_width = bullet\_size[0]  bullet\_height = bullet\_size[1]  bullet\_x\_pos = random.randint(0, screen\_width - bullet\_width)  bullet\_y\_pos = 0  bullet\_speed = 5 | pygame.init()  bullets = []  A\_images = []  B\_images = []  spawn\_bullet\_time = pygame.time.get\_ticks()  spawn\_bullet\_interval = 3000  korean='C:/Users/chosun/Desktop/dodge-the-bullets-main/korean.ttf'  # 폰트 설정  font = pygame.font.Font(korean, 30)  # 디스플레이에 시간 띄우는 함수  def display\_time(elapsed\_time):      elapsed\_time\_str = str(timedelta(seconds=elapsed\_time))      time\_text = font.render(elapsed\_time\_str, True, (255, 255, 255))      screen.blit(time\_text, (10, 10))  # 화면의 크기  screen\_width = 960  # 가로 크기  screen\_height = 556  # 세로 크기  screen = pygame.display.set\_mode((screen\_width, screen\_height))  # 이게 화면 크기다  # FPS  clock = pygame.time.Clock()  # 배경 이미지  background = pygame.image.load(r'C:\Users\chosun\Desktop\dodge-the-bullets-main\background.png')  # 엔딩 이미지  ending\_image = pygame.image.load(r'C:\Users\chosun\Desktop\dodge-the-bullets-main\졸업.png')  #배드엔딩 이미지  bad\_ending\_image=pygame.image.load(r'C:\Users\chosun\Desktop\dodge-the-bullets-main\학사경고.png')  # 솔져 스프라이트  sol = pygame.image.load(r"C:\Users\chosun\Desktop\dodge-the-bullets-main\sol.png")  sol\_size = sol.get\_rect().size  sol\_width = sol\_size[0]  sol\_height = sol\_size[0]  sol\_x\_pos = (screen\_width / 2) - (sol\_width / 2)  sol\_y\_pos = screen\_height - sol\_height  # 솔져의 속도  to\_x = 0  sol\_speed = 10  # F 학점  bullet\_image = pygame.image.load(r"C:\Users\chosun\Desktop\dodge-the-bullets-main\bullet.png")  bullet\_size = bullet\_image.get\_rect().size  bullet\_width = bullet\_size[0]  bullet\_height = bullet\_size[1]  bullet\_speed = 2  #A학점  A\_image = pygame.image.load(r"C:\Users\chosun\Desktop\dodge-the-bullets-main\A.png")  A\_size = A\_image.get\_rect().size  A\_width = A\_size[0]  A\_height = A\_size[1]  A\_speed = 3  #B학점  B\_image = pygame.image.load(r"C:\Users\chosun\Desktop\dodge-the-bullets-main\B.png")  B\_size = B\_image.get\_rect().size  B\_width = B\_size[0]  B\_height = B\_size[1]  B\_speed = 4  # 스코어 정의  score = 0  score\_time = pygame.time.get\_ticks()  score\_interval = 10000  # 10000밀리초마다 점수 상승 구간  # 타임 정의  start\_time = pygame.time.get\_ticks()  # 게임시작 할때 초기 시간값  spawn\_bullet\_time = start\_time  # 시간 측정을 위한 변수  spawn\_A\_time = start\_time  A\_interval = 15000  spawn\_B\_time = start\_time  B\_interval = 12500 |

보시면 기본 배경화면 추가

총알 추가

스코어 ,시간 추가

그리고 한글을 사용하기 위해 한글폰트 추가등 여러가지를 추가하느라 구문이 달라졌습니다

**실행구문**

|  |  |
| --- | --- |
| 원본 | 내가 만든 프로젝트 |
| running = True #게임이 진행중인가요? true  while running:      dt = clock.tick(60) #게임 화면의 초당 프레임 수      for event in pygame.event.get(): #게임이 진행중일때 어떤 이벤트가 발생하면 여기서 처리 (ex.마우스, 키보드 클릭킹 등등)          if event.type == pygame.QUIT: #창이 닫히는 이벤트가 발생하였는가?              running = False #창이 닫힌다 = 게임도 진행이 멈춘다.          if event.type == pygame.KEYDOWN: #키를 눌렀을때              if event.key == pygame.K\_LEFT: #왼쪽을 누르면 왼쪽으로                  to\_x -= sol\_speed              elif event.key == pygame.K\_RIGHT: #오른쪽을 누르면 오른쪽으로                  to\_x += sol\_speed          if event.type == pygame.KEYUP: #아무것도 누르지 않으면 걍 가자미!!              if event.key == pygame.K\_LEFT or event.key == pygame.K\_RIGHT:                  to\_x = 0      sol\_x\_pos  += to\_x      #솔의 탈영막기 (가로막기)      if sol\_x\_pos < 0:          sol\_x\_pos = 0      elif sol\_x\_pos > screen\_width - sol\_width:          sol\_x\_pos = screen\_width - sol\_width      bullet\_y\_pos += bullet\_speed #총알 떨어지기      #총알 떨어지기      if bullet\_y\_pos > screen\_height:          bullet\_y\_pos = 0          bullet\_x\_pos = random.randint(0, screen\_width - bullet\_width)      #충돌처리 불쌍한 솔,,,      sol\_rect = sol.get\_rect()      sol\_rect.left = sol\_x\_pos      sol\_rect.top = sol\_y\_pos      #총알의 충돌처리      bullet\_rect = bullet.get\_rect()      bullet\_rect.left = bullet\_x\_pos      bullet\_rect.top = bullet\_y\_pos | while running:      dt = clock.tick(60)  # 게임 화면의 초당 프레임 수      current\_time = pygame.time.get\_ticks()      elapsed\_time = (current\_time - start\_time) // 1000      #움직이는 이벤트      for event in pygame.event.get():          if event.type == pygame.QUIT: #창이 닫히는 이벤트가 발생하였는가?              running = False #창이 닫힌다 = 게임도 진행이 멈춘다.          if event.type == pygame.KEYDOWN: #키를 눌렀을때              if event.key == pygame.K\_LEFT: #왼쪽을 누르면 왼쪽으로                  to\_x -= sol\_speed              elif event.key == pygame.K\_RIGHT: #오른쪽을 누르면 오른쪽으로                  to\_x += sol\_speed          if event.type == pygame.KEYUP: #아무것도 누르지 않으면 안가는 코드 !!              if event.key == pygame.K\_LEFT or event.key == pygame.K\_RIGHT:                  to\_x = 0       # A 이미지 떨어뜨리기      # 시간마다 생성되는 A      if current\_time - spawn\_A\_time >= A\_interval:          spawn\_A\_time = current\_time      # 새로운 A 이미지          new\_A = {              'x\_pos': random.randint(0, screen\_width - A\_width),              'y\_pos': 0,              'speed': A\_speed          }          A\_images.append(new\_A)      # 시간마다 생성되는 B      if current\_time - spawn\_B\_time >= B\_interval:          spawn\_B\_time = current\_time      # 새로운 A 이미지          new\_B = {              'x\_pos': random.randint(0, screen\_width - B\_width),              'y\_pos': 0,              'speed': B\_speed          }          B\_images.append(new\_B)      # 시간마다 생성되는 총알      if current\_time - spawn\_bullet\_time >= spawn\_bullet\_interval:          spawn\_bullet\_time = current\_time          bullet\_speed += 0.2          print(f"Bullet speed increased to {bullet\_speed}")          # 새로운 총알          new\_bullet = {              'x\_pos': random.randint(0, screen\_width - bullet\_width),              'y\_pos': 0,              'speed': bullet\_speed          }          bullets.append(new\_bullet)      # 기존총알 업그레이드      for bullet in bullets:          bullet['y\_pos'] += bullet['speed']          # 결과값          if sol\_rect.colliderect(pygame.Rect(bullet['x\_pos'], bullet['y\_pos'], bullet\_width, bullet\_height)):              print("F 이미지를 획득하였습니다!")              score -= 5.0              print(f"현재 스코어: {score}")              bullets.remove(bullet)      # 기존 A 이미지 업그레이드      for A in A\_images:          A['y\_pos'] += A['speed']          if sol\_rect.colliderect(pygame.Rect(A['x\_pos'], A['y\_pos'], A\_width, A\_height)):              print("A 이미지를 획득하였습니다!")              score += 4.5              print(f"현재 스코어: {score}")                A\_images.remove(A)      # 기존 B 이미지 업그레이드      for B in B\_images:          B['y\_pos'] += B['speed']          if sol\_rect.colliderect(pygame.Rect(B['x\_pos'], B['y\_pos'], B\_width, B\_height)):              print("B 이미지를 획득하였습니다!")              score += 3.5              print(f"현재 스코어: {score}")              B\_images.remove(B)        sol\_x\_pos += to\_x      #화면 밖으로 못나가는 학생      if sol\_x\_pos < 0:          sol\_x\_pos = 0      elif sol\_x\_pos > screen\_width - sol\_width:          sol\_x\_pos = screen\_width - sol\_width      sol\_rect = sol.get\_rect()      sol\_rect.left = sol\_x\_pos      sol\_rect.top = sol\_y\_pos        # 삭제 구문      bullets = [bullet for bullet in bullets if bullet['y\_pos'] <= screen\_height]        # A삭제 구문      A\_images = [A for A in A\_images if A['y\_pos'] <= screen\_height]      # B삭제 구문      B\_images = [B for B in B\_images if B['y\_pos'] <= screen\_height] |

보시면 총알이 떨어지는 개수 추가

스피드 추가

삭제구문 추가

스코어 추가

등 기본 실행 화면 차이를 여러가지를 추가하였습니다

**엔딩 구문**

|  |  |
| --- | --- |
| 원본화면 | 내가 만든 프로젝트 |
| if sol\_rect.colliderect(bullet\_rect):          print ("당신은 정말 저질이군요 당신 때문에 쟤 인생은 망했스빈다")          running = False      screen.blit(background, (0, 0)) #배경의 위치 (aka 좌표)      screen.blit(sol, (sol\_x\_pos, sol\_y\_pos)) #솔의 위치      screen.blit(bullet, (bullet\_x\_pos, bullet\_y\_pos)) #총알의 좌표      pygame.display.update()  #게임이 종료하면 pygame도 종료  pygame.quit() | if current\_time - score\_time >= score\_interval:          score\_time = current\_time          score += 1          print(f"Score increased to {score}")      screen.blit(background, (0, 0))      screen.blit(sol, (sol\_x\_pos, sol\_y\_pos))      #140학점을 이수해야 졸업 가능      if  score > 3:          elapsed\_time = (pygame.time.get\_ticks() - start\_time) // 1000          ending\_start\_time = pygame.time.get\_ticks()          screen.blit(ending\_image, (0, 0))          # 가운데에 졸업을 축하합니다 메시지 표시          text\_surface = font.render("졸업을 축하합니다!", True, (0, 0, 0))          text\_surface1 = font.render("140학점이 넘어 졸업했습니다.", True, (0, 0, 0))          text\_rect = text\_surface.get\_rect(center=(screen\_width // 2, screen\_height // 2))          text\_rect1 = text\_surface1.get\_rect(center=(screen\_width // 2+25, screen\_height // 2+25))          screen.blit(text\_surface, text\_rect)          screen.blit(text\_surface1, text\_rect1)          # 엔딩 이미지를 표시할 시간이 지나면 게임 종료          if pygame.time.get\_ticks() - ending\_start\_time > ending\_display\_time:  break      elif score<0:           # bed 엔딩 이미지 표시              screen.blit(bad\_ending\_image, (0, 0))              text\_surface = font.render("학사경고", True, (0, 0, 0))              text\_surface1 = font.render("당신은 졸업하지 못했습니다 !", True, (0, 0, 0))              text\_rect = text\_surface.get\_rect(center=(screen\_width // 2, screen\_height // 2))              text\_rect1 = text\_surface1.get\_rect(center=(screen\_width // 2+25, screen\_height // 2+25))              screen.blit(text\_surface, text\_rect)              screen.blit(text\_surface1, text\_rect1)              # bed 엔딩이 표시될 때 ending\_start\_time을 업데이트              ending\_start\_time = pygame.time.get\_ticks()              # ending\_display\_time이 지날 때까지 대기 후 종료              if pygame.time.get\_ticks() - ending\_start\_time > ending\_display\_time:                 break      else:      # 총알 여러개 소환          for bullet in bullets:              screen.blit(bullet\_image, (bullet['x\_pos'], bullet['y\_pos']))      # A여러개 소환          for A in A\_images:              screen.blit(A\_image, (A['x\_pos'], A['y\_pos']))      # B여러개 소환          for B in B\_images:              screen.blit(B\_image, (B['x\_pos'], B['y\_pos']))      # 기존 A,B 이미지 업그레이드 이후에 삭제 구문을 따로 두어 remove 에러를 방지          A\_images = [A for A in A\_images if A['y\_pos'] <= screen\_height]          B\_images = [B for B in B\_images if B['y\_pos'] <= screen\_height]      # 스코어 띄워놓는 구문          score\_text = font.render(f"학점: {score}", True, (255, 255, 255))          screen.blit(score\_text, (screen\_width - 150, 10))      display\_time(elapsed\_time)      pygame.display.update()      pygame.quit() |

10점마다 1점씩 추가되는 구문

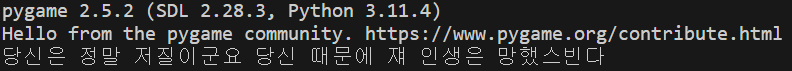
보시면 총알 여러개 소환구문 엔딩조건이 스코어를 기준으로 바꾸고 엔딩은 디버그파일이 아닌 새로운 화면에서 출력되도록 바꾸었습니다

**4. 실행 결과**

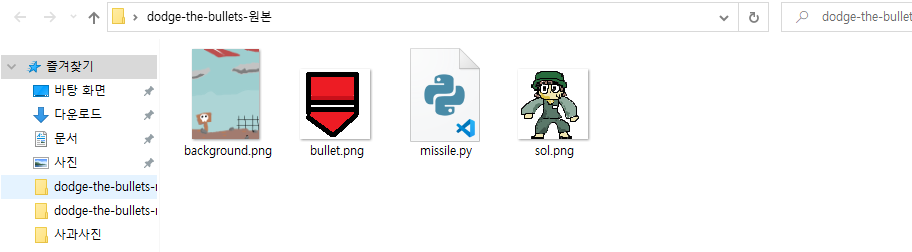
실행 결과에 대한 설명과 함께 이미지 첨부 필수입니다.

원본실행화면 

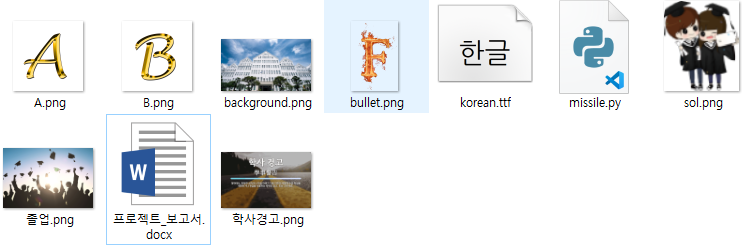
결과화면



원본 리소스팩



수정 리소스팩



수정실행화면



수정결과화면

* 졸업 엔딩



* 학사경고 엔딩

