"DS/AI 프로그래밍"

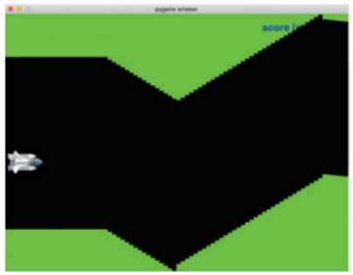
8주차 PyGame (Cave 게임 구현)

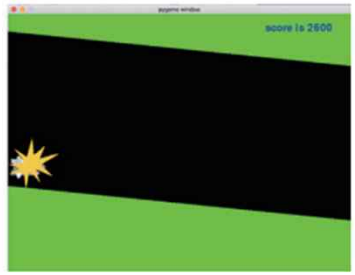
J.-K.- Seo
데이터과학원

KUIDS
고려대학교 데이터과학원

Chapter
01 Cave

간단한 가로 스크롤 게임입니다. 스페이스 키를 누르면 윗 방향으로 가속도가 붙습니다. 동굴은 점차 좁아집니다.





소스 코드(cave.py)

```
""" cave - Copyright 2016 Kenichiro Tanaka """
import sys
from random import randint
import pygame
from pygame.locals import QUIT, Rect, KEYDOWN, K_SPACE

pygame.init()
pygame.key.set_repeat(5, 5)
```

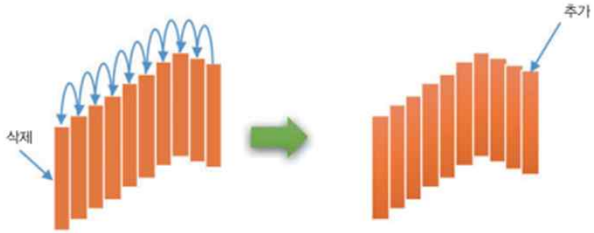
```
SURFACE = pygame.display.set_mode((800, 600))
FPSCLOCK = pygame.time.Clock()

def main():
    """ 메인 루틴 """
    walls = 80
    ship_y = 250
    velocity = 0
    score = 0
    slope = randint(1, 6)
    sysfont = pygame.font.SysFont(None, 36)
    ship_image = pygame.image.load("ship.png")
    bang_image = pygame.image.load("bang.png")
    holes = []
    for xpos in range(walls):
        holes.append(Rect(xpos * 10, 100, 10, 400))
    game_over = False

    while True:
        is_space_down = False
        for event in pygame.event.get():
            if event.type == QUIT:
                pygame.quit()
                sys.exit()
            elif event.type == KEYDOWN:
                if event.key == K_SPACE:
                    is_space_down = True

        # 내 캐릭터를 이동
        if not game_over:
            score += 10
            velocity += -3 if is_space_down else 3
            ship_y += velocity

            # 동굴을 스크롤
            edge = holes[-1].copy()
            test = edge.move(0, slope)
            if test.top <= 0 or test.bottom >= 600:
                slope = randint(1, 6) * (-1 if slope > 0 else 1)
```


KUIDS
고려대학교 데이터과학원

```
                edge.inflate_ip(0, -20)
            edge.move_ip(10, slope)
            holes.append(edge)
            del holes[0]
            holes = [x.move(-10, 0) for x in holes]

            # 충돌?
            if holes[0].top > ship_y or \
                holes[0].bottom < ship_y + 80:
                game_over = True

        # 그리기
        SURFACE.fill((0, 255, 0))
        for hole in holes:
            pygame.draw.rect(SURFACE, (0, 0, 0), hole)
        SURFACE.blit(ship_image, (0, ship_y))
        score_image = sysfont.render("score is {}".format(score),
                                     True, (0, 0, 225))
        SURFACE.blit(score_image, (600, 20))

        if game_over:
            SURFACE.blit(bang_image, (0, ship_y-40))

        pygame.display.update()
        FPSCLOCK.tick(15)

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

1 : 개요

75줄 정도의 간단한 게임입니다. 많은 직사각형을 뿌리면서 가로 방향으로 나열해서 동굴을 표현하고 있습니다. 프레임마다 모든 직사각형을 왼쪽 방향으로 움직이고, 맨 앞(왼쪽 끝)의 직사각형을 제거, 오른쪽 끝에 새로운 직사각형을 추가해 가로 방향의 스크롤을 구현하였습니다.



2 : 전역 변수

이번 게임에서 전역 변수는 SURFACE(윈도)와 FPSCLOCK(프레임 레이트 조정용의 타이머) 두 개입니다. 다음은 전역 코드입니다.

```
pygame.init()
pygame.key.set_repeat(5, 5)
SURFACE = pygame.display.set_mode((800, 600))
FPSCLOCK = pygame.time.Clock()
```

pygame.init()으로 pygame을 초기화합니다. pygame.key.set_repeat()는 키의 반복 기능을 설정하는 pygame의 메서드입니다. 키를 눌렀을 때 연속해서 KEYDOWN 이벤트를 생성하기 위해서 호출합니다. set_mode((800, 600))으로 화면 크기를 설정하고, FPSCLOCK 객체를 만듭니다.

KUIDS
고려대학교 데이터과학원

main()

이 게임의 유일한 함수입니다. 길이가 길어서 나눠 설명합니다. 주요 지역 변수를 다음에 나열합니다.

walls	동굴을 구성하는 직사각형의 수
ship_y	내 캐릭터의 Y 좌표
velocity	내 캐릭터가 상하로 이동할 때의 속도
score	점수
slope	동굴의 기울기(옆의 직사각형과 Y 축 방향으로 얼마나 비켜 있는지)
holes	동굴을 구성하는 직사각형을 저장하는 배열
game_over	게임 오버인지 아닌지 여부의 플래그

다음 코드로 동굴을 구성하는 직사각형을 작성합니다.

```
for xpos in range(walls):
    holes.append(Rect(xpos * 10, 100, 10, 400))
```

Rect는 pygame 안에 정의된 클래스입니다. 인수는 (X 좌표, Y 좌표, 폭, 높이)입니다. X 축 방향으로 10씩 비키면서 직사각형을 walls개 만들고 있습니다. 만든 직사각형은 리스트 holes에 추가해 갑니다.

초기화가 끝나면 while True:로 메인 루프에 진입합니다. 루프를 시작할 때마다, is_space_down을 False로 초기화합니다. 다음에 이벤트 큐에서 이벤트를 취득하고 QUIT이면 게임을 종료합니다. 이벤트 유형이 KEYDOWN, 키 코드가 K_SPACE이면 is_space_down을 True로 설정합니다.

다음 코드로 내 캐릭터를 이동합니다.

```
        if not game_over:
            score += 10
            velocity += -3 if is_space_down else 3
            ship_y += velocity
```

먼저 if not game_over: 에서 게임 오버가 아닐 때(게임 중)의 처리를 기술합니다. 점수를 10 증가하

KUIDS
고려대학교 데이터과학원

고, 스페이스 키 입력 상태에 따라서 속도를 −3 (상승), 또는 +3 (하강) 변화시킵니다.

동굴의 스크롤은 다음 코드입니다.

```
edge = holes[-1].copy()
test = edge.move(0, slope)
if test.top <= 0 or test.bottom >= 600:
    slope = randint(1, 6) * (-1 if slope > 0 else 1)
    edge.inflate_ip(0, -20)
edge.move_ip(10, slope)
holes.append(edge)
del holes[0]
holes = [x.move(-10, 0) for x in holes]
```

edge = holes[walls − 1].copy()에서는 오른쪽 끝의 직사각형을 복사해서 변수 edge에 저장합니다. 배열 번호는 0부터 시작됩니다. 따라서 walls−1로 마지막 요소를 취득할 수 있습니다. 사실은 holes[−1]이라고 기술해도 마지막 요소를 취득할 수 있습니다. 그러므로 이 줄은 다음과 같이 바꿔도 똑같이 동작합니다.

```
edge = holes[-1].copy()
```

다음으로 새로 만든 직사각형을 이동시켜서 천장이나 바닥에 부딪히지 않는지 검출합니다. 부딪쳤을 때는 동굴의 기울기를 반대 방향으로 해야 합니다.

```
test = edge.move(0, slope)
```

여기서 move는 Rect를 이동하는 메서드입니다. edge를 Y 축 방향으로 slope만큼 움직입니다. 이때 edge는 변화하지 않고, 새로운 장소로 이동한 직사각형 test가 반환되는 것에 주의하세요. 처음에 move_ip이 아니라 move를 사용한 것은 만일 이동해서 충돌하는지 아닌지를 검출하기 위해서입니다. 다음의 if 문으로 천장 또는 바닥에 닿았는지 판정합니다.

```
if test.top <= 0 or test.bottom >= 600:
```

충돌했을 때는 방향을 바꿔서 동굴의 크기를 한층 작게 줄입니다. 방향을 바꾸는 것이 다음 코드입니다. 기울기와 부호 반전으로 분할해서 생각하면 이해하기 쉽습니다.

```
slope = randint(1, 6) * (-1 if slope > 0 else 1)
```
기울기의 절대값을 난수로 생성 — 기울기의 부호를 반전

edge.inflate_ip(0, −20)으로 Y 축 방향의 크기를 20만큼 작게 합니다. 다음으로 edge.move_ip(10, slope)로 오른쪽 끝의 직사각형을 X 축 방향으로 +10, Y 축 방향으로 slope만큼 이동합니다. 이번에는 move가 아닌 move_ip를 사용해 스스로 이동하는 것에 주의하세요. 나머지는 다음의 순서로 가로 스크롤을 실행합니다.

맨 끝(오른쪽 끝)에 추가	holes.append(edge)
맨 앞의 직사각형을 삭제	del holes[0]
전체를 10 왼쪽으로 이동	holes = [x.move(-10, 0) for x in holes]

다음 코드로 내 캐릭터가 동굴 벽에 충돌했는지를 판정합니다.

```
# 충돌?
if holes[0].top > ship_y or holes[0].bottom < ship_y + 80:
    game_over = True
```

ship_y는 내 캐릭터의 Y 좌표(위쪽 끝)입니다. 아래쪽 끝은 ship_y+80으로 했습니다. 이 값을 조정하면 충돌 판정을 엄격하게 하거나 완만하게 합니다. 이러한 값이 동굴의 왼쪽 끝의 직사각형 holes[0] 범위에 들어가 있는지를 조사하는 것입니다.

나머지는 그리기입니다.

```
        SURFACE.fill((0, 255, 0))
        for hole in holes:
            pygame.draw.rect(SURFACE, (0, 0, 0), hole)
        SURFACE.blit(ship_image, (0, ship_y))
        score_image = sysfont.render("score is {}".format(score),
                                     True, (0, 0, 225))
        SURFACE.blit(score_image, (600, 20))

        if game_over:
            SURFACE.blit(bang_image, (0, ship_y-40))
```

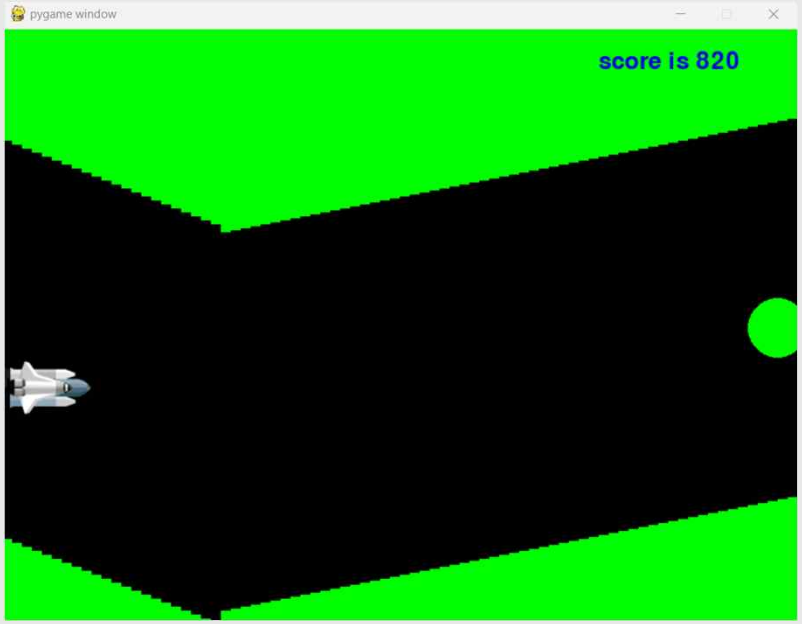
전체 화면을 녹색으로 칠하고, 동굴 구멍의 직사각형을 그리고, 내 캐릭터와 점수를 나타냅니다. 게임오버 시에는 그 메시지를 표시합니다.

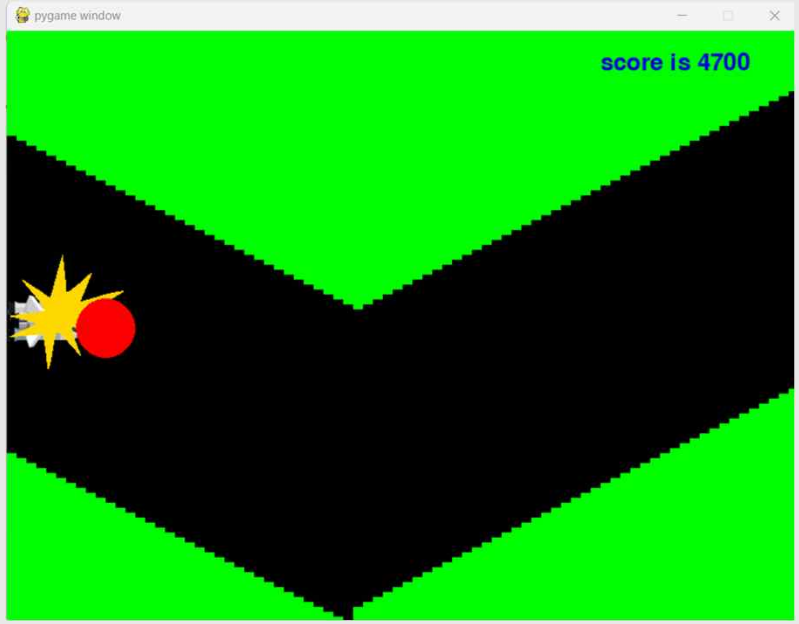
마지막으로 pygame.display.update()로 그리기를 화면에 반영하고, 타이머를 사용해서 FPS를 조정합니다.

설명은 이상입니다. 동굴의 변화하는 정도에 삼각함수를 사용하면 더욱 매끄러운 동굴이 될 것입니다. 도중에 장애물을 생성해도 재미있겠죠.

KUIDS
고려대학교 데이터과학원

연습문제:
장애물(Obstalce)이 출현하도록 구현해 본다.





KUIDS
고려대학교 데이터과학원

The End