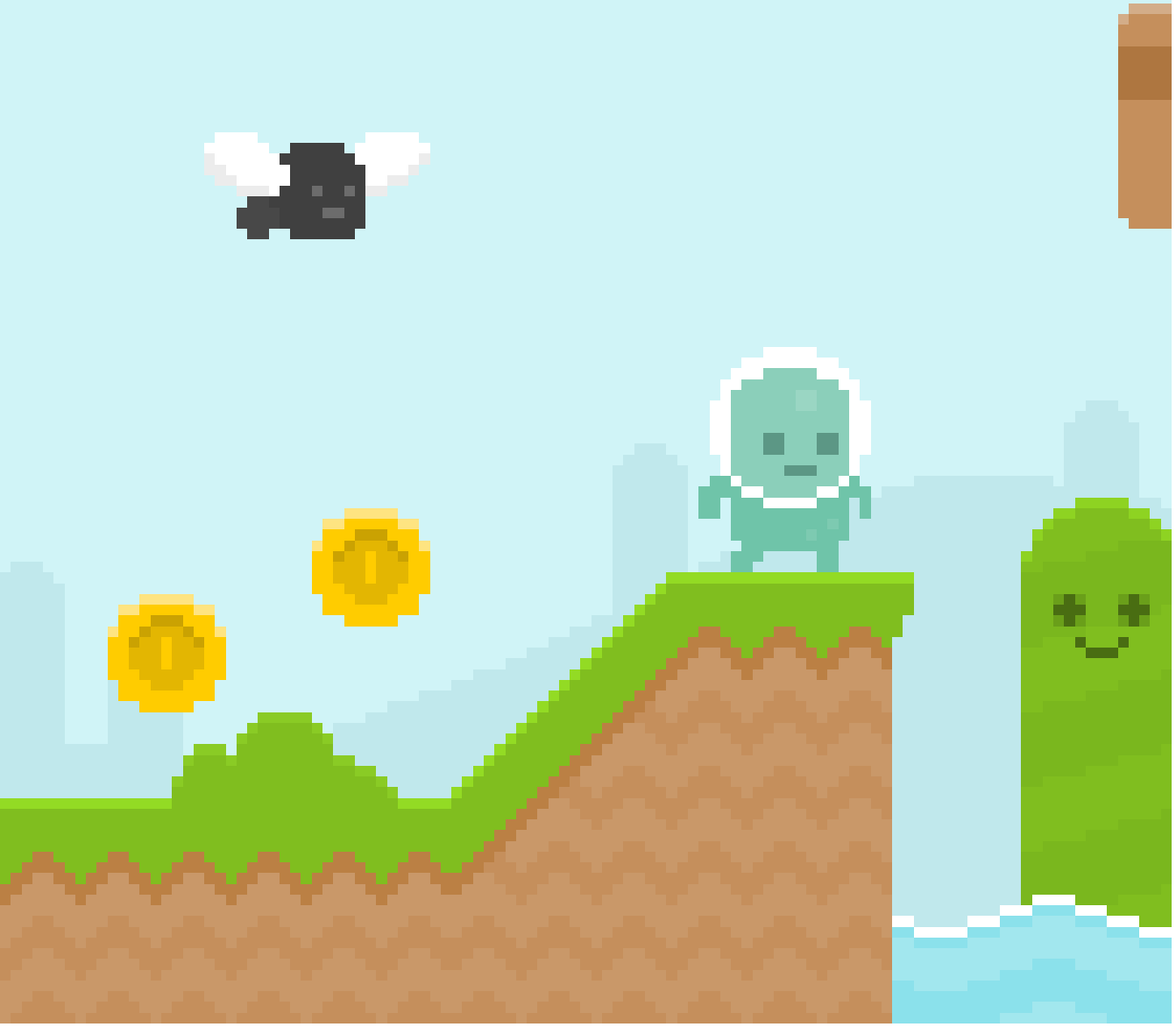


게임을 만들며 배우는 파이썬 3 (with pygame)

- 이벤트, 점수 다루기

오류중학교 코딩 동아리
강사: 여 은 선

확인합시다!

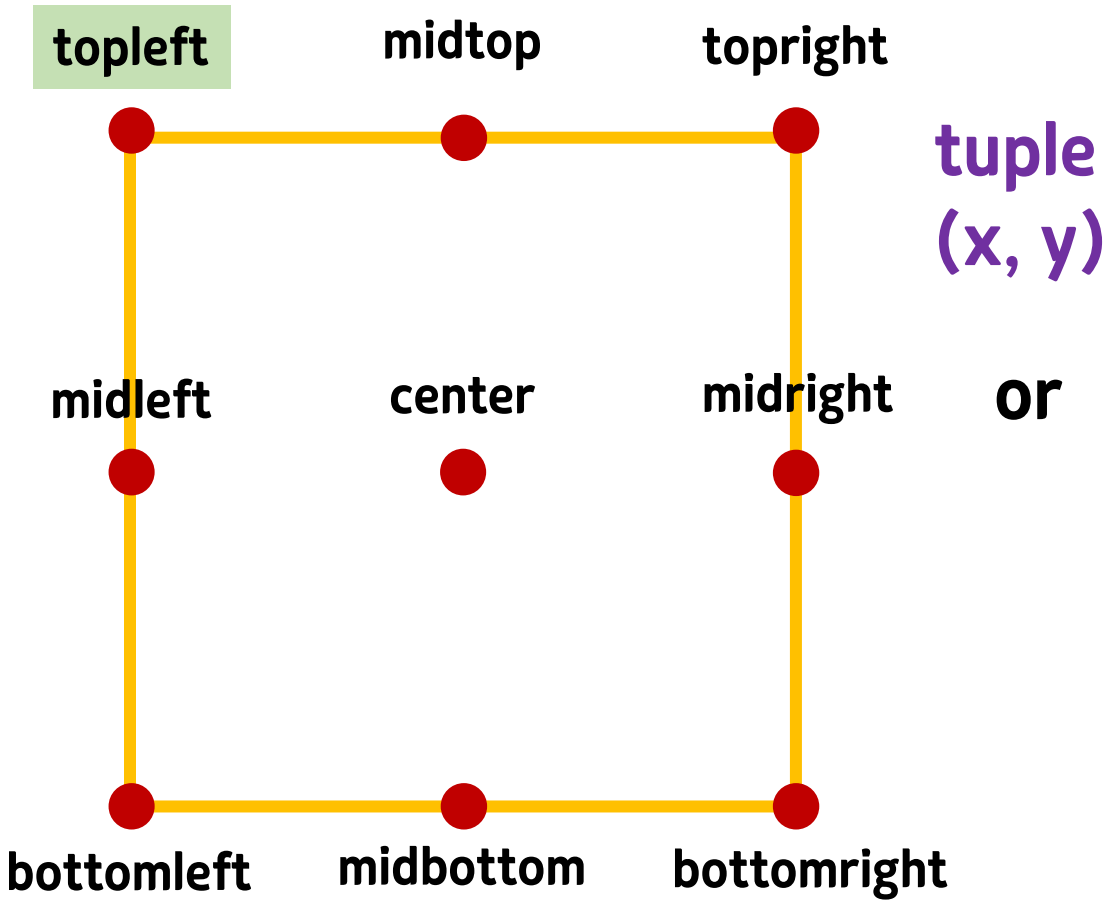
☆ https://github.com/chpink0518/oryuCC.git
소스 코드를 다운로드합니다

☆ infos3.ipynb 파일만 make_game폴더에서 카피해 옵니다
main.py는 2강에서 사용한 것에 더해 사용합니다

☆ infos3.ipynb 파일의 각 셀 단위를 main.py로 옮겨오며 테스트
-> 주의: 순서나 위치가 맞지 않음, 셀을 실행하지 마세요!!

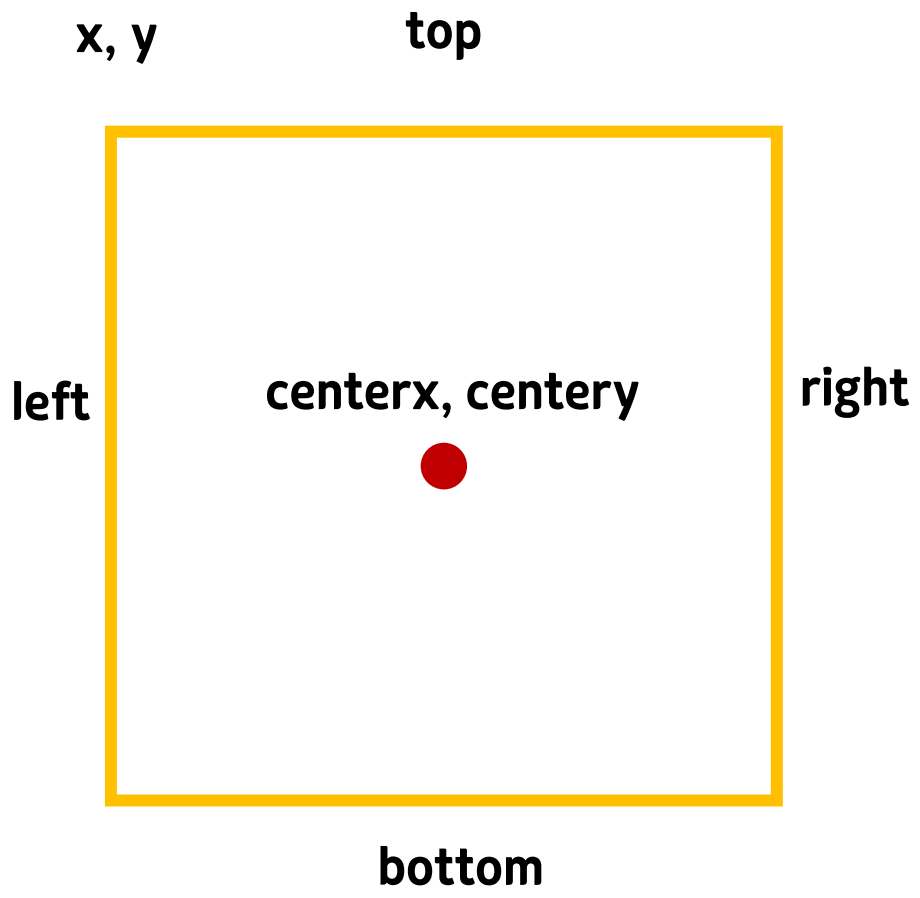
Rectangle - 사각형

☆ 이미지 변환 서피스를 화면에 그리는 기준



tuple
(x, y)

or



캐릭터들을 땅으로

☆ 플레이어가 땅을 디디고 서게 해 봅시다

```
player_rect = player_surf.get_rect(bottomleft = (80, 300))
```

-> 땅 서피스의 top좌표인 300 사용

☆ infos3.ipynb의 1번째 셀

```
# 01. snail_surface with rectangle
snail_rect = snail_surface.get_rect(bottomright = (600, 300))

snail_rect.x -= 4
if snail_rect.right <= 0: snail_rect.left = 800

win.blit(snail_surface, snail_rect)
```

-> snail_x_pos 삭제
대신 snail_rect 연계 좌표 사용

-> 보다 정교한 처리 가능

○ bottomright 를 기준점으로 잡은 이유는?

○ 플레이어 뒤로 달팽이가 지나가는 이유는?

뭘 준비를 합시다

☆ infos3.ipynb의 3번째 셀

```
#03. make event loop code for jump
if event.type == pygame.KEYDOWN:
    if event.key == pygame.K_SPACE:
        print('점프')

if event.type == pygame.MOUSEBUTTONDOWN:
    if event.button == 1:
        if player_rect.collidepoint(event.pos):
            print('점프')
```

-> 어떤 키가 눌렸을 때
-> 그 키가 스페이스 키면

-> 어떤 마우스 버튼이 눌렸을 때
-> 그 버튼이 왼쪽 버튼이라면
-> 플레이어의 사각형이
마우스 위치와 충돌한다면

+ event.button:

1:	왼쪽 버튼 클릭
2:	가운데 버튼 클릭
3:	오른쪽 버튼 클릭
4:	스크롤 업
5:	스크롤 다운

가짜지만 좀 더 그럴 듯하게

☆ infos3.ipynb의 4번째 셀

```
#04. add gravity
player_gravity = 0

player_gravity += 1
player_rect.y += player_gravity
```

-> 매 프레임마다 가속도 증가
증가한 가속도 만큼 플레이어 y값 증가
: 오래 떨어질수록 빠른 속도

☆ infos3.ipynb의 5번째 셀

```
# 05. event loop re-visit
player_gravity = -20
```

-> 이전에 추가한 loop 부분에서
print('점프')를 이것으로 대체

☆ infos3.ipynb의 6번째 셀

```
#06. set y boundary for the player
if player_rect.bottom > 300:
    player_rect.bottom = 300
```

-> 플레이어가 땅을 뚫고 내려가지 않게

2단 점프 막기

☆ infos3.ipynb의 8번째 셀

```
#07. player only jump once at a time
if player_rect.bottom == 300:
```

☆ mission!

이 비교문을 사용해서
한번에 한 번만 점프할 수 있도록 만드세요

게임 오버 상태

☆ infos3.ipynb의 8번째 셀

```
#08. game over state
gameActive = True

if gameActive:
    # 화면 업데이트 내용
    # collision
    if snail_rect.colliderect(player_rect):
        gameActive = False

else:
    win.fill((94, 129, 162))
```

-> 충돌 시 게임 오버
게임 오버 화면 출력

이벤트 루프 수정

☆ infos3.ipynb의 9번째 셀

```
#09. change event loop also
startTime = 0

if gameActive:
    # do event check here
else:
    if event.type == pygame.KEYDOWN and event.key == pygame.K_SPACE:
        gameActive = True
        snail_rect.left = 800
        startTime = pygame.time.get_ticks()
```

-> 게임 오버 상태에서 스페이스 바를 누르면
다시 게임이 액티브 상태로 돌아옴

-> 달팽이 사각형 좌표를 바꾸는 이유는?

버틴 시간을 스코어로

☆ infos3.ipynb의 10번째 셀

```
#10. get time for score
score = 0

def display_score():
    c_time = int((pygame.time.get_ticks() - startTime)/1000)
    score_surf = my_font.render(f'SCORE: {c_time}', False, (64, 64, 64))
    score_rect = score_surf.get_rect(center = (400, 50))
    win.blit(score_surf, score_rect)
    return c_time

score = display_score()
```

-> 함수 정의 후 호출

-> pygame.time.get_ticks()
게임 경과 시간 체크(밀리세컨드)

-> 게임 오버 후 되돌아 가면 스코어는?

게임 오버 화면을 장식할 서피스 생성

☆ infos3.ipynb의 11번째 셀

```
#11. make surface for game over screen
player_stand = pygame.image.load('graphics/player/player_stand.png').convert_alpha()
player_stand = pygame.transform.rotozoom(player_stand, 0, 2)
player_stand_rect = player_stand.get_rect(center = (400, 200))

game_name = my_font.render('Pixel Runner', False, (111, 196, 169))
game_name_rect = game_name.get_rect(center = (400, 80))
game_msg = my_font.render('Press space to run', False, (111, 196, 169))
game_msg_rect = game_msg.get_rect(center = (400, 325))
```

-> pygame.transform.rotozoom(대상, 회전각도, 확대 배수)
값을 바꿔 테스트해 보자

생성한 서피스들을 그리자

☆ infos3.ipynb의 12번째 셀

```
#12. blit game over screen
else:
    win.fill((94, 129, 162))
    win.blit(game_name, game_name_rect)
    win.blit(player_stand, player_stand_rect)

    score_msg = my_font.render(f'Your score: {score}', False, (111, 196, 169))
    score_msg_rect = score_msg.get_rect(center = (400, 325))

    if score > 0: win.blit(score_msg, score_msg_rect)
    else: win.blit(game_msg, game_msg_rect)
```

-> 다른 서피스와 달리 스코어 메시지 서피스를 이곳에서 만드는 이유는?

타이머 이벤트를 만들자

☆ infos3.ipynb의 13번째 셀

```
#13. create a custom timer event
obstacle_timer = pygame.USEREVENT
pygame.time.set_timer(obstacle_timer, 1500)

# event loop
if event.type == obstacle_timer :
    print('test')
```

-> 타이머 간격: ms
- 1.5초

-> pygame event 수 : 32개
이중 23개를 pygame이 이미 사용 중
24부터 32까지 유저 이벤트 생성 가능(9개)

-> 이후 이벤트 추가 시 pygame.USEREVENT + 1과
같이 값을 증가해서 사용하면 된다

더 많은 달팽이들

☆ infos3.ipynb의 14번째 셀

```
#14. change snail movement
from random import randint

# remove snail_rect & collision detection part
obstacle_rect_list = []

# event loop
if event.type == obstacle_timer :
    obstacle_rect_list.append(snail_surface.get_rect(bottomright = (randint(900, 1100), 300)))
```

-> 랜덤한 정수를 구하기: randint

-> 서피스는 하나, rect(위치-랜덤)는 여러 개
- 화면을 벗어난 위치에서 x 좌표 생성
- y 좌표는 땅에 붙어서 이동하도록 고정

더 많은 달팽이들

☆ infos3.ipynb의 14번째 셀

```
# define the obstacle movement function
def obstacle_movement(obs_list):
    if obs_list:  # list is not null
        for obs_rect in obs_list:
            obs_rect.x -= 5
            win.blit(snail_surface, obs_rect)
        obs_list = [obst for obst in obs_list if obst.right > 0]
    return obs_list

# comment snail movement block
# obstacle movement - main loop
obstacle_rect_list = obstacle_movement(obstacle_rect_list)
```

-> 화면을 벗어난 rect는 지워준다

-> 리스트의 모든 x 좌표값을 5만큼 뺀 위치에 달팽이를 그린다
-> 화면을 벗어난 rect는 지워준다
-> obs_list는 지역 변수이므로 return값으로 반환해
전역 변수인 obstacle_rect_list에 대입해 준다

파리의 등장

☆ infos3.ipynb의 15번째 셀

```
#15. add different enemies - fly
fly_surf = pygame.image.load('graphics/fly/fly1.png').convert_alpha()

if event.type == obstacle_timer and gameActive:
    if randint(0, 2):
        obstacle_rect_list.append(snail_surface.get_rect(bottomright = (randint(900, 1100), 300)))
    else:
        obstacle_rect_list.append(fly_surf.get_rect(bottomright = (randint(900, 1100), 180)))
```

-> 파리 서피스 생성

-> 0,1 중 랜덤한 값을 구해 1이면 달팽이, 0이면 파리
- 파리는 공중에 떠 있으므로 y 좌표가 180

파리의 등장

☆ infos3.ipynb의 15번째 셀

```
def obstacle_movement(obs_list):
    if obs_list:  # list is not null
        for obs_rect in obs_list:
            obs_rect.x -= 5
            if obs_rect.bottom == 300: win.blit(snail_surface, obs_rect)
            else: win.blit(fly_surf, obs_rect)

        obs_list = [obst for obst in obs_list if obst.right > 0]
    return obs_list
```

-> bottom 값에 따라 어느 서피스를 그릴지 결정

충돌 감지 방법 수정하기

☆ infos3.ipynb의 16번째 셀

```
#16. fix the collision logic

def is_Collide(plyr_rect, ob_rect_list):
    if ob_rect_list:
        for obs_rect in ob_rect_list:
            if plyr_rect.colliderect(obs_rect): return False
    return True

# main loop
gameActive = is_Collide(player_rect, obstacle_rect_list)

# game over
obstacle_rect_list.clear()
```