**등장인물:**

- **코드쌤:** 착하고, 재미있고, 똑똑한 코딩 선생님
- **지혜:** 호기심 많고 질문하기 좋아하는 학생
- **슬기:** 직관적이고 만들기를 좋아하는 학생

---

**코드쌤:** 지혜야, 슬기야! 오늘 우리가 함께 탐험할 세상은 바로 '파이썬으로 게임 만들기'란다! 게임 만드는 거, 생각만 해도 신나지 않니?

**지혜:** 와! 게임이요? 제가 맨날 하는 그 게임을 직접 만들 수 있는 거예요?

**슬기:** 대박! 그럼 막 캐릭터도 움직이고, 점수도 나오고 그래요?

**코드쌤:** 물론이지! 하지만 모든 위대한 여정에는 첫걸음이 있는 법! 우리가 게임을 만들려면 특별한 도구 상자가 필요한데, 그게 바로 Pygame이라는 라이브러리야. 지난 시간에 `pip install pygame`으로 다들 설치는 잘 했지? 마치 요리하기 전에 필요한 재료와 도구를 장바구니에 담는 것과 같아.

**지혜:** 네! 설치는 했는데, 그 다음엔 뭘 해야 하나요? 오늘 배울 내용에 `pygame.init()`이라는 게 있던데요!

**코드쌤:** 오, 예리한데 지혜! 맞아. `pygame.init()`! 자, 상상해 보자. 우리가 로봇을 조립해서 게임을 만들려고 해. Pygame이라는 로봇 부품 상자를 열었어. 이 부품들을 바로 쓸 수 있을까?

**슬기:** 음… 전원을 켜야 하지 않을까요? 아니면 부품들을 서로 연결한다거나?

**코드쌤:** 빙고, 슬기! `pygame.init()`은 바로 그거야. Pygame이라는 멋진 로봇(게임 개발 도구들)의 **전원을 켜고, 모든 부품들이 서로 '안녕! 이제부터 같이 일하자!' 하고 인사하면서 준비 태세를 갖추는 과정**이라고 생각하면 돼. 그래픽 카드야, 준비됐니? 사운드 카드야, 너도? 키보드 입력 받을 준비는? 이렇게 물어보고 "네!" 대답을 받는 거지.

**지혜:** 아하! 그래서 `pygame.init()`을 안 하면 Pygame 기능들을 쓸 수 없거나, 로봇이 제대로 움직이지 않는 것처럼 에러가 날 수 있겠네요?

**코드쌤:** 정확해, 지혜! Pygame을 사용하기 전에는 반드시, 아주 반드시 `pygame.init()`을 호출해서 모든 모듈과 장비들을 깨워줘야 해. 마치 아침에 일어나서 "오늘 하루도 파이팅!" 하고 기지개를 켜는 것과 같지!

**(코드 조각 1: 초기화와 종료)**

```python
import pygame

# Pygame 로봇, 일어나! 모든 준비를 시작해!
pygame.init()
print("Pygame 로봇, 성공적으로 깨어났습니다!")

# 자, 오늘은 여기까지! Pygame 로봇, 이제 쉬어도 돼!
pygame.quit()
print("Pygame 로봇, 안전하게 잠들었습니다!")
```

**슬기:** 와, pygame.init() 하니까 "성공적으로 깨어났습니다!" 라고 뜨는 것 같아요! 그럼 pygame.quit()은 뭐예요? 로봇을 다시 재우는 건가?

**코드쌤:** 맞아, 슬기! 아주 훌륭한 비유야! 게임이 끝나면 우리가 사용했던 모든 Pygame 기능들, 예를 들어 화면 창, 소리 장치 등을 **깔끔하게 정리하고 "이제 끝났으니 다들 퇴근하세요!" 하고 알려주는 역할**을 해. 만약 pygame.quit()을 안 하고 그냥 프로그램을 꺼버리면, 로봇이 일하다 말고 갑자기 전원이 나가버리는 거랑 비슷해. 사용했던 물건들이 정리되지 않아서 다음에 문제가 생길 수도 있거든.

**지혜:** 아, 그래서 pygame.init()으로 시작하고, pygame.quit()으로 끝내는 게 한 쌍이군요! 마치 문을 열고 들어갔다가, 나갈 때 문을 닫는 것처럼요!

**코드쌤:** 완벽한 이해다, 지혜야! 자, 그럼 이제 본격적으로 게임의 '무대'를 만들어 볼까?

**(코드 조각 2: 게임 창 만들기)**

```
import pygame

# 1. Pygame 로봇, 일어나!
pygame.init()

# 창의 크기를 정해볼까? 가로 800, 세로 600 픽셀!
WINDOW_WIDTH = 800
WINDOW_HEIGHT = 600

# 2. 자, 이제 이 크기로 게임 무대(창)를 만들어줘!
window = pygame.display.set_mode((WINDOW_WIDTH, WINDOW_HEIGHT))
# 잠깐! 여기서 (WINDOW_WIDTH, WINDOW_HEIGHT) 이렇게 괄호가 두 개인 이유가 뭘까요?
# set_mode는 크기 정보를 '하나의 묶음(튜플)'으로 받기 때문이란다! (800, 600) 이렇게 말이지.

# 3. 우리 게임 무대에 멋진 이름표도 달아주자!
pygame.display.set_caption("나의 첫 게임 무대!")

# 4. 잠깐! 이대로 실행하면 창이 번쩍! 하고 사라질 거야.
# 왜냐하면 컴퓨터는 우리가 시킨 일을 너무 빨리 끝내버리거든.
# 그래서 창이 계속 열려 있도록 '마법의 주문'을 걸어줘야 해!
# 그건 바로 '게임 루프(Game Loop)'라는 건데, 이건 다음 시간에 더 자세히!
# 일단은 창이 1초만 보였다가 사라지도록 해볼까?
pygame.time.delay(1000) # 1000밀리초 = 1초 동안 잠시 멈춤

# 5. Pygame 로봇, 이제 정리하고 쉬자!
pygame.quit()
```

**슬기:** 와! pygame.display.set_mode((800, 600)) 하니까 진짜로 네모난 창이 생겼어요! WINDOW_WIDTH랑

WINDOW_HEIGHT는 창 크기를 정하는 숫자군요!

**코드쌤:** 맞아! pygame.display는 Pygame 로봇 중에서 **화면 표시 담당 로봇**이라고 생각하면 돼. 그리고 set_mode()는 그 화면 담당 로봇에게 "이봐, 가로 800, 세로 600짜리 스크린 좀 준비해줘!" 하고 명령하는 거지. 그 결과로 만들어진 스크린(창)을 우리는 window라는 이름표를 붙여서 기억하는 거야.

**지혜:** pygame.display.set_caption("나의 첫 게임 무대!")는 창 위에 뜨는 제목을 정하는 거 맞죠? 제 이름으로 바꿔도 돼요? "지혜의 게임" 이렇게요!

**코드쌤:** 당연하지! 얼마든지! 아주 창의적인걸? 자, 그런데 아까 코드에서 창이 금방 사라졌지? 게임은 계속 진행되어야 하잖아. 마치 연극이 시작되면 배우들이 계속 연기를 하는 것처럼 말이야.

**(코드 조각 3: 꺼지지 않는 창과 이벤트 처리)**

```
import pygame

pygame.init()

WINDOW_WIDTH = 800
WINDOW_HEIGHT = 600
window = pygame.display.set_mode((WINDOW_WIDTH, WINDOW_HEIGHT))
pygame.display.set_caption("지혜와 슬기의 게임 월드!")

# 게임이 계속 실행 중인지 알려주는 깃발!
running = True

# 게임 루프: 이 안에서 게임의 모든 일이 계속 반복해서 일어난다!
while running:
    # 어떤 이벤트가 발생했는지 Pygame 로봇이 계속 귀를 기울이고 있어!
    # (마우스 클릭, 키보드 누르기, 창 닫기 버튼 누르기 등등)
    for event in pygame.event.get(): # 발생한 이벤트 목록을 하나씩 가져와서 확인
        print(event) # 어떤 이벤트가 발생하는지 한번 보자! (창을 움직이거나 키를 눌러봐!)

        # 만약 발생한 이벤트가 "창 닫기" 버튼을 누른 것이라면?
        if event.type == pygame.QUIT:
            running = False # 깃발을 내려서 게임 루프를 멈추자!

# 게임 루프가 끝났으니, 이제 정리 시간!
pygame.quit()
```

**슬기:** 와! while running: 하니까 창이 안 꺼져요! 계속 떠 있어요! 그런데 print(event) 하니까 뭔가 엄청 많이 나와요! 마우스 움직일 때마다 숫자가 바뀌고…

**코드쌤:** 잘 봤어, 슬기! while running: 이게 바로 '**게임 루프**'야. running이라는 깃발이 올라가 있는 동안에는

이 while문 안의 코드가 계속해서, 아주 빠르게 반복 실행되는 거지. 마치 영화 필름이 계속 돌아가는 것처럼!

**지혜:** pygame.event.get()은 뭐예요? event라는 건 사건이라는 뜻인데… 아! 우리가 마우스를 움직이거나, 키보드를 누르거나, 창의 X 버튼을 누르는 모든 행동이 '이벤트'인 건가요?

**코드쌤:** 천재인데, 지혜! 정확해! pygame.event.get()은 **"지금 혹시 무슨 일 없었니?"** 하고 Pygame 로봇에게 물어보는 거야. 그럼 로봇은 "네, 방금 마우스가 왼쪽으로 10만큼 움직였어요!" 또는 "방금 'A' 키가 눌렸어요!" 아니면 "창 닫기 버튼이 눌렸어요!" 하고 발생한 사건(이벤트)들을 쭉 알려줘.

**슬기:** 그래서 for event in pygame.event.get(): 이렇게 해서 로봇이 알려준 사건들을 하나씩 살펴보는 거군요! 그리고 if event.type == pygame.QUIT: 이건 "만약 사건의 종류가 '창 닫기(QUIT)'라면" 이라는 뜻이고요!

**코드쌤:** 완벽해! pygame.QUIT은 Pygame이 미리 정해놓은 "창 닫기 버튼 눌림" 이벤트의 이름표(상수)야. 그래서 그 이벤트가 발생하면, 우리는 running 깃발을 False로 바꿔서 "이제 그만 돌아도 돼!" 하고 게임 루프를 멈추는 거지.

**지혜:** 아하! pygame.init()으로 Pygame을 깨우고, pygame.display.set_mode()로 창을 만들고, set_caption()으로 제목을 달고, while running 게임 루프로 창을 계속 유지하면서, pygame.event.get()으로 사용자 행동(이벤트)을 감지하고, pygame.QUIT 이벤트가 발생하면 루프를 빠져나와 pygame.quit()으로 마무리하는 거군요!

**코드쌤:** 이야, 지혜랑 슬기, 오늘 정말 대단한데! Pygame의 가장 기본적이면서도 핵심적인 뼈대를 완벽하게 이해했어! 마치 게임을 만들기 위한 설계도를 얻은 것과 같단다.

**슬기:** 선생님! 그럼 이제 저 창에다가 그림도 그리고, 캐릭터도 넣을 수 있는 거예요?

**코드쌤:** 물론이지! 오늘 배운 건 게임의 무대를 만들고, 관객(사용자)의 반응을 살피는 방법이었어. 다음 시간에는 이 무대 위에 멋진 그림을 그리고, 색칠하는 방법을 배워볼 거야! 기대되지?

**지혜/슬기:** 네! 완전 기대돼요!

**코드쌤:** 좋아! 오늘 정말 수고 많았고, 너희들의 반짝이는 눈빛을 보니 선생님도 정말 신이 난다! 다음 시간에 더 재미있는 내용으로 만나자!