코드쌤: 지혜야, 슬기야! 우리가 지난 시간에는 '게임 지휘관' Game 클래스가 등장해서 게임 전체의 정보를 관리하고, '수배 몬스터' 미션도 내주는 걸 봤었지?

지혜: 네! Game 클래스가 점수판도 그려주고, 찾아야 할 몬스터도 알려줘서 진짜 게임다워졌어요! 📜👾

슬기: 맞아요! 근데 선생님, 플레이어가 몬스터랑 부딪히면 어떻게 되는 거예요? 수배 몬스터를 잡으면 점수를 얻고, 다른 몬스터랑 부딪히면 목숨이 깎이나요? 그리고 목숨을 다 쓰면 정말 게임 오버가 되는 건가요? 😱

코드쌤: 오호! 슬기가 드디어 게임의 가장 핵심적인 재미, 바로 **'상호작용과 결과'**에 대한 궁금증을 가졌네! 오늘은 바로 그 Game 클래스 안에 숨겨진 **'충돌 처리(Collision Check)'** 마법과, 게임의 흐름을 바꾸는 **'새로운 라운드 시작', '게임 일시 정지', '게임 초기화'** 같은 놀라운 기능들을 자세히 살펴볼 거야! 마치 게임의 규칙을 만들고 심판을 보는 것과 같단다!

(코드 조각 1: Game 클래스의 새로운 능력들 - 충돌을 감지하고 게임을 지휘하라!)

```
# ... (pygame.init(), WINDOW 설정, FPS, clock 등은 이전과 동일) ...

class Game():
  # __init__ 메소드에 monster_group도 전달받도록 바뀌었네!
  def __init__(self, player, monster_group):
    self.score = 0
    self.round_number = 0 # 라운드 번호는 0부터 시작 (start_new_round에서 1이 됨)
    self.player = player
    self.round_time = 0
    self.frame_count = 0
    self.monster_group = monster_group # 몬스터 그룹도 Game 객체가 알고 있게 됨!

    self.font = pygame.font.Font("CrotahFreeVersionItalic-z8Ev3.ttf", 24)
    # (wanted monster 관련 코드는 이전과 유사하므로 설명은 핵심 변경점 위주로!)
    blue_monster = pygame.image.load("blue_monster.png")
    # ... (monster_images, wanted_monster_type, image, rect 설정은 동일) ...
    self.monster_images = [blue_monster, pygame.image.load("green_monster.png"), pygame.image.load("orang
    self.wanted_monster_type = random.randint(0, 3)
    self.wanted_monster_image = self.monster_images[self.wanted_monster_type]
    self.wanted_monster_rect = self.wanted_monster_image.get_rect()
    self.wanted_monster_rect.centerx = WINDOW_WIDTH/2
    self.wanted_monster_rect.top = 35

  def update(self): # Game 객체의 update는 매 프레임마다 호출!
    self.frame_count += 1
    if self.frame_count == FPS: # (원래 코드엔 60이지만, FPS를 사용하는 게 더 좋겠죠?)
      self.round_time += 1
```

```
        self.frame_count = 0

    # --- 여기가 오늘의 하이라이트 중 하나! 충돌 검사 실행! ---
    self.check_collision() # 매 프레임마다 "혹시 플레이어랑 몬스터 부딪혔니?" 확인!

def draw(self): # (draw 메소드는 이전과 거의 동일: 점수, 라운드, 목숨 등 표시)
    # ... (생략) ...
    pass # 실제로는 이전 코드의 draw 내용이 여기에!

# --- 충돌 감지 및 처리 메소드! ---
def check_collision(self):
    # "플레이어(self.player)와 몬스터 그룹(self.monster_group)에 속한 몬스터 중
    #  단 하나라도 충돌한 몬스터가 있니?" (있으면 그 몬스터를 알려줘!)
    collided_monster = pygame.sprite.spritecollideany(self.player, self.monster_group)

    if collided_monster: # 만약 어떤 몬스터와 충돌했다면?
        # 1. 충돌한 몬스터가 "수배 몬스터"와 같은 종류인가?
        if collided_monster.type == self.wanted_monster_type:
            self.monster_group.remove(collided_monster) # 잡았으니 몬스터 그룹에서 제거!
            self.score += 100 # 점수 100점 획득!
            self.player.success_sound.play() # "성공!" 소리 재생! (Player 객체가 소리를 가짐)
            # 만약 화면에 아직 다른 몬스터들이 남아있다면?
            if self.monster_group: # 몬스터 그룹이 비어있지 않다면
                self.choose_new_wanted() # 새로운 "수배 몬스터"를 다시 뽑자!
            # 화면의 모든 몬스터를 다 잡았다면? (몬스터 그룹이 비었다면)
            else:
                self.player.reset() # 플레이어 위치 초기화!
                self.start_new_round() # 다음 라운드 시작!

        # 2. 엉뚱한 몬스터와 부딪혔다면?
        else:
            self.player.die_sound.play() # "실패/죽음" 소리 재생!
            self.player.lives -= 1       # 목숨 1 감소!
            self.player.reset()          # 플레이어 위치 초기화!
            # 목숨을 다 썼는지 확인!
            if self.player.lives <= 0:
                # 게임 오버! 게임 일시 정지하고 메시지 보여주기!
                self.pause_game(f"Final Score: {self.score}", "Press 'Enter' to play again")
```

```
# --- 새로운 "수배 몬스터"를 고르는 메소드! ---
def choose_new_wanted(self):
  # 현재 화면에 남아있는 몬스터들 중에서 아무나 하나를 골라 새로운 수배범으로!
  # list(self.monster_group)으로 그룹을 리스트로 바꿔야 random.choice 사용 가능!
  new_wanted_monster = random.choice(list(self.monster_group))
  self.wanted_monster_type = new_wanted_monster.type # 그 몬스터의 종류로 업데이트!
  self.wanted_monster_image = new_wanted_monster.image # 그 몬스터의 이미지로 업데이트!

# --- 새로운 라운드를 시작하는 메소드! ---
def start_new_round(self):
  self.round_number += 1 # 라운드 번호 1 증가!
  self.round_time = 0    # 라운드 시간 초기화!
  self.frame_count = 0   # 프레임 카운트 초기화!

  # 현재 라운드 번호만큼! 각 종류의 몬스터들을 새로 생성해서 그룹에 추가!
  # 라운드가 올라갈수록 몬스터 수가 점점 많아지겠네! (난이도 상승!)
  for _ in range(self.round_number): # _ 변수는 반복 횟수만 필요하고 값은 안 쓸 때 사용
    # (원래 코드에서는 i를 썼지만, _가 더 명확할 수 있어요)
    # 4종류의 몬스터를 각각 라운드 수만큼 추가 (즉, 1라운드엔 4마리, 2라운드엔 8마리...)
    self.monster_group.add(Monster(random.randint(0, WINDOW_WIDTH - 64), random.randint(101, WINDOW_HEI
    self.monster_group.add(Monster(random.randint(0, WINDOW_WIDTH - 64), random.randint(101, WINDOW_HEI
    self.monster_group.add(Monster(random.randint(0, WINDOW_WIDTH - 64), random.randint(101, WINDOW_HEI
    self.monster_group.add(Monster(random.randint(0, WINDOW_WIDTH - 64), random.randint(101, WINDOW_HEI

  self.choose_new_wanted() # 새로운 라운드 시작 시 수배 몬스터도 새로 뽑아야지! (원래 코드엔 없지만, 이

# --- 게임을 일시 정지하고 메시지를 보여주는 메소드! ---
def pause_game(self, main_text, sub_text):
  global running # 게임 전체를 멈추거나 재시작하려면 running 변수를 바꿔야 하니 global 선언!
  WHITE = (255, 255, 255)
  # (main_text, sub_text 만들고 그리는 부분은 이전과 유사)
  # ... (생략) ...
  main_text_render = self.font.render(main_text, True, WHITE) # 변수명 변경
  main_rect = main_text_render.get_rect()
  main_rect.center = (WINDOW_WIDTH/2, WINDOW_HEIGHT/2)

  sub_text_render = self.font.render(sub_text, True, WHITE) # 변수명 변경
  sub_rect = sub_text_render.get_rect()
  sub_rect.center = (WINDOW_WIDTH/2, WINDOW_HEIGHT/2 + 40)
```

```
    window_surface.fill((0,0,0)) # 화면을 검게 칠하고
    window_surface.blit(main_text_render, main_rect) # 메시지 표시
    window_surface.blit(sub_text_render, sub_rect)
    pygame.display.update() # 화면 업데이트!

    is_paused = True
    while is_paused: # "일시 정지" 루프! 키 입력을 기다림
      for event in pygame.event.get():
        if event.type == pygame.QUIT: # 창 닫으면 완전히 종료
          is_paused = False
          running = False
        elif event.type == pygame.KEYDOWN: # 키가 눌렸는데
          if event.key == pygame.K_RETURN: # 그게 '엔터' 키라면?
            is_paused = False # "일시 정지" 루프를 끝내고
            self.reset_game() # 게임을 초기 상태로 리셋!

  # --- 게임 전체를 초기 상태로 되돌리는 메소드! ---
  def reset_game(self):
    self.score = 0 # (원래 코드엔 socre 오타!) 점수 0점!
    self.round_number = 0 # 라운드도 0부터 (start_new_round에서 1이 됨)

    self.player.lives = STARTING_PLAYER_LIVES # (상수가 있다면 좋겠죠? 여기선 임의로 5)
    self.player.safe = 3 # (상수가 있다면 좋겠죠? 여기선 임의로 3)
    self.player.reset() # 플레이어 위치도 처음으로!

    # 모든 몬스터를 없애고 새로 시작해야 하니, 몬스터 그룹을 비워주자!
    self.monster_group.empty() # (이 부분이 중요!)
    self.start_new_round() # 그리고 새로운 라운드(사실상 새 게임) 시작!
```

지혜: Game 클래스의 __init__을 보니까 이제 monster_group도 전달받네요! 그럼 Game 객체가 플레이어뿐만 아니라 화면에 있는 모든 몬스터들의 정보도 알 수 있게 되는 거군요!

코드쌤: 정확해! Game 클래스가 게임 세상의 거의 모든 것을 알고 있어야, "플레이어가 어떤 몬스터랑 부딪혔는지" 같은 중요한 판단을 내릴 수 있겠지? 그리고 Game 클래스의 update() 메소드 안에서 self.check_collision()이라는 새로운 함수를 매 프레임마다 호출하고 있어. 이게 바로 오늘 이야기의 핵심 중 하나야!

슬기: check_collision() 메소드 안을 보니까 collided_monster = pygame.sprite.spritecollideany(self.player, self.monster_group) 라는 처음 보는 함수가 있어요! 이건 뭐예요? spritecollideany라니, 뭔가 스프라이트끼리 부딪혔는지 확인하는 건가?

코드쌤: 빙고! pygame.sprite.spritecollideany()는 아주 똑똑한 마법 주문이야. **"첫 번째 인자로 준**

스프라이트(여기서는 self.player)가, 두 번째 인자로 준 스프라이트 그룹(여기서는 self.monster_group) 안에 있는 어떤 스프라이트와 단 하나라도 충돌했니?" 라고 물어보는 거야. 만약 충돌했다면, 그 충돌한 몬스터 객체를 collided_monster에 담아주고, 아무와도 충돌하지 않았다면 None(아무것도 없음)을 돌려줘.

지혜: 와! 그럼 if collided_monster: 이 조건문은 "만약 플레이어가 어떤 몬스터랑 부딪혔다면?" 이라는 뜻이네요! 그리고 그 안에서 if collided_monster.type == self.wanted_monster_type: 이렇게 해서 "그 부딪힌 몬스터가 혹시 우리가 찾던 수배 몬스터니?" 라고 확인하는 거고요!

코드쌤: 정확하게 이해했어! 만약 수배 몬스터를 제대로 잡았다면, 그 몬스터를 화면에서 없애고(self.monster_group.remove(co 점수를 크게 주고, "성공!" 소리도 나지! 그리고 만약 화면에 다른 몬스터들이 아직 남아있다면 self.choose_new_wanted()를 호출해서 새로운 수배 몬스터를 다시 뽑아. 만약 모든 몬스터를 다 잡았다면? self.start_new_round()를 호출해서 다음 라운드로 넘어가는 거야!

슬기: 만약 엉뚱한 몬스터랑 부딪히면 "실패!" 소리가 나고 목숨이 하나 줄어드네요! 😱 그리고 플레이어 위치도 self.player.reset()으로 처음 자리로 돌아가고요! 만약 목숨을 다 쓰면(self.player.lives <= 0) self.pause_game(...) 함수가 호출돼요! 이건 게임이 멈추는 건가?

코드쌤: 맞아! pause_game() 메소드는 게임을 잠시 멈추고 "최종 점수"와 "다시 하려면 엔터 키를 누르세요" 같은 메시지를 화면에 보여줘. 그리고 그 안에서 while is_paused: 라는 루프가 돌면서 플레이어가 엔터 키를 누르기를 기다린단다. 엔터 키가 눌리면? self.reset_game() 함수를 호출해서 게임의 모든 상태를 깨끗하게 처음으로 되돌리고 다시 시작하는 거지!

지혜: start_new_round() 메소드를 보니까 라운드 번호가 올라갈수록 for _ in range(self.round_number): 이렇게 해서 몬스터를 더 많이 만들어내네요! 1라운드에는 종류별로 1마리씩(총 4마리), 2라운드에는 종류별로 2마리씩(총 8마리)! 점점 어려워지겠어요!

코드쌤: 그게 바로 이 게임의 재미 중 하나지! 라운드가 진행될수록 도전 과제가 늘어나는 거야. 그리고 reset_game() 메소드에서는 점수, 라운드, 플레이어 목숨, 위치뿐만 아니라 self.monster_group.empty()를 호출해서 이전 라운드의 모든 몬스터를 깨끗하게 없애주는 것도 중요해. 그렇지 않으면 이전 라운드 몬스터들이 새 라운드에도 계속 남아있을 테니까!

(Player 클래스의 새로운 점들) 코드쌤: 자, 그리고 우리 Player 클래스에도 몇 가지 새로운 점들이 생겼어!

```
class Player(pygame.sprite.Sprite):
  def __init__(self):
    # ... (이전 image, rect, velocity 설정은 동일) ...
    self.lives = 5 # 처음 목숨은 5개!
    self.safe = 3  # '안전' 기회는 3번!

    # --- 플레이어도 이제 소리를 직접 갖게 되었네! ---
    self.success_sound = pygame.mixer.Sound("success.mp3")
    self.die_sound = pygame.mixer.Sound("die.mp3")
    self.safe_zone_sound = pygame.mixer.Sound("safe_zone.mp3") # (safe_zone.mp3 파일 필요!)
```

```
    # update 메소드는 동일

    # --- 플레이어 위치를 처음으로 되돌리는 메소드! ---
    def reset(self):
      self.rect.centerx = WINDOW_WIDTH/2
      self.rect.bottom = WINDOW_HEIGHT

    # --- "안전지대"로 순간이동하는 (가상의) 메소드! ---
    # (이 safe_zone 메소드는 현재 게임 루프에서 스페이스바를 누르면 호출되도록 되어있네!)
    def safe_zone(self):
      if self.safe > 0: # '안전' 기회가 남아있다면
        self.safe -= 1  # 기회 1번 사용!
        self.safe_zone_sound.play() # "안전지대!" 효과음!
        self.rect.bottom = WINDOW_HEIGHT # 플레이어를 화면 맨 아래로 (안전한 곳으로 가정?)
```

슬기: Player 클래스 안에 self.lives랑 self.safe 변수가 생겼고, success_sound, die_sound 같은 소리들도 Player가 직접 가지고 있네요! 그럼 Game 클래스에서 self.player.success_sound.play() 이렇게 플레이어의 소리를 재생시킬 수 있겠어요!

지혜: reset() 메소드는 플레이어 위치를 처음으로 돌려주는 거고, safe_zone() 메소드는 self.safe 기회가 남아있을 때 플레이어를 화면 아래로 옮겨주나 봐요! 게임 루프에서 스페이스바를 누르면 이 safe_zone()이 호출되도록 되어 있네요! 위험할 때 쓰는 비상 탈출 기능인가 봐요!

(게임 루프와 객체 생성 부분의 변경점) 코드쌤: 맞아! 그리고 메인 코드에서 Game 객체를 만들 때 my_game = Game(my_player, my_monster_group) 이렇게 my_monster_group도 전달해주고, 게임 시작 전에 my_game.start_new_round()를 호출해서 첫 번째 라운드의 몬스터들을 미리 준비시키는 것도 중요한 변경점이야. 그리고 게임 루프 안에서 스페이스바를 누르면 my_player.safe_zone()을 호출하는 이벤트 처리도 추가됐지.

슬기: 와! 오늘 정말 많은 걸 배웠어요! spritecollideany로 충돌을 확인하고, 그 결과에 따라 점수를 주거나 목숨을 깎고, 새로운 라운드를 시작하거나 게임을 다시 시작하는 것까지! 진짜 게임 하나가 어떻게 만들어지는지 본 것 같아요!

코드쌤: 훌륭해! 오늘 우리는 Game 클래스가 어떻게 게임의 전체 흐름을 지휘하고, 플레이어와 몬스터의 상호작용을 처리하며, 게임의 상태를 바꾸는지 아주 자세히 살펴봤어. 이 정도면 너희도 이제 복잡한 게임의 구조를 이해하고 직접 만들어볼 수 있는 멋진 게임 개발자가 된 거나 다름없단다!

지혜: 이 코드를 바탕으로 저만의 규칙을 더 추가해보고 싶어요! 예를 들면 특정 시간 안에 수배 몬스터를 못 잡으면 목숨이 깎인다거나, 특별한 아이템이 나와서 플레이어에게 도움을 주는 거요!

코드쌤: 아주 멋진 아이디어야, 지혜! 오늘 배운 이 게임의 구조를 잘 활용하면 너희들의 상상력을 마음껏 펼쳐서 더욱 재미있고 개성 넘치는 게임을 만들 수 있을 거야! 너희들의 다음 작품이 정말 기대되는걸!