코드쌤: 자, 우리 멋진 탐험가 지혜와 슬기! 지난 시간에 `pygame.init()`으로 Pygame 로봇을 깨우고, `pygame.display.set_mode()`로 게임 무대(창)를 만들고, 게임 루프로 창을 계속 열어두는 법까지 배웠죠?

지혜: 네, 선생님! 제 이름으로 창 제목도 바꿨어요! "지혜의 게임 월드!" 라고요! 😊

슬기: 맞아요! X 버튼 누르면 착하게 꺼지도록 `pygame.QUIT` 이벤트도 처리했고요! 이제 저 하얀 도화지 같은 창에 뭔가 그리고 싶어요!

코드쌤: 아주 좋은 질문이야, 슬기! 오늘 바로 그 비어있는 무대를 알록달록하게 꾸며볼 거란다! 마치 화가가 캔버스에 그림을 그리듯이 말이야. 자, 그림을 그리려면 가장 먼저 뭐가 필요할까?

지혜: 음… 물감이요! 여러 가지 색깔 물감!

코드쌤: 정답! 컴퓨터 세상에서도 색깔을 표현하는 방법이 있단다. 바로 **RGB**라는 약속인데, **R**ed(빨강), **G**reen(초록), **B**lue(파랑) 이 세 가지 빛의 색을 섞어서 모든 색을 만들어내는 거야. 마치 빛의 삼원색처럼!

(코드 조각 1: 색깔 정의하기)

```
#Initialize Pygame
pygame.init()

#Create a window and set its caption
WINDOW_WIDTH = 600
WINDOW_HEIGHT = 600

window = pygame.display.set_mode((WINDOW_WIDTH, WINDOW_HEIGHT))
pygame.display.set_caption("알록달록 그림 그리기!")

#Define colors (R, G, B) - 각 색은 0부터 255까지의 숫자로 표현해요!
BLACK = (0, 0, 0)          # 빨강0, 초록0, 파랑0 = 검은색!
WHITE = (255, 255, 255)    # 빨강 최대, 초록 최대, 파랑 최대 = 하얀색!
RED = (255, 0, 0)        # 빨강만 최대!
GREEN = (0, 255, 0)      # 초록만 최대!
BLUE = (0, 0, 255)       # 파랑만 최대!
YELLOW = (255, 255, 0)   # 빨강 + 초록 = 노랑!
CYAN = (0, 255, 255)     # 초록 + 파랑 = 청록!
MAGNETA = (255, 0, 255)  # 빨강 + 파랑 = 자홍색!
```

슬기: 와! `BLACK = (0, 0, 0)` 이고 `WHITE = (255, 255, 255)` 네요! 숫자가 클수록 밝아지는 건가요?

코드쌤: 정확해, 슬기! 각 색깔마다 0부터 255까지의 세기로 빛을 섞는다고 생각하면 돼. 모두 0이면 빛이 없으니 검은색, 모두 255면 모든 빛이 강하게 합쳐져서 하얀색이 되는 거지. `RED = (255, 0, 0)`은 빨간색 빛만 최대, 나머지는 0인 거야.

지혜: 그럼 `YELLOW = (255, 255, 0)`은 빨간 빛이랑 초록 빛을 최대로 섞고, 파란 빛은 안 섞은 거네요? 신기하다! 빛을 섞으면 노란색이 되는군요!

코드쌤: 맞아! 물감 섞는 거랑은 조금 다르지? 빛의 세계는 그렇단다. 자, 이제 이 예쁜 색깔들로 우리 게임 무대의 배경색부터 칠해볼까?

(코드 조각 2: 배경색 채우기 및 화면 업데이트의 비밀!)

```
# ... (이전 색깔 정의 코드는 그대로) ...

# 게임 무대(window) 전체를 파란색(BLUE)으로 채워줘!
window.fill(BLUE)
# 앗! 이대로 실행하면 파란색이 안 보일 수도 있어! 왜일까?

# ... (게임 루프 코드는 아래에) ...
```

슬기: 어? 선생님, `window.fill(BLUE)` 하고 실행했는데 창이 그냥 하얘요! 파란색으로 안 바뀌는데요? 버그인가요? 😱

코드쌤: 하하, 슬기야, 버그는 아니란다! 아주 중요한 비밀이 숨어있어. 우리가 컴퓨터한테 "배경을 파란색으로 칠해!" 라고 명령(`window.fill(BLUE)`)하면, 컴퓨터는 열심히 도화지(메모리 속 화면 공간)에 파란색을 칠해. 그런데! 그 칠한 도화지를 우리 눈앞의 스크린에 "짠! 다 그렸어요!" 하고 보여주는 과정이 한 번 더 필요해. 그게 바로 `pygame.display.update()` 란다!

지혜: 아! 지난 시간에 배운 `pygame.display.set_mode()`는 무대를 만드는 거였고, `pygame.display.update()`는 그 무대에 그린 걸 관객에게 보여주는 거군요! 마치 연극 중간중간 "자, 보세요!" 하고 커튼을 잠깐 열어주는 것 같아요!

코드쌤: 비유 천재인데, 지혜! 맞아. 우리가 `window.fill()`이나 `pygame.draw` 같은 명령으로 그림을 그리면, 그건 아직 Pygame 내부의 '작업용 스케치북'에 그려지는 거야. 이걸 실제 화면에 "반영!" 시키려면 `pygame.display.update()` 또는 `pygame.display.flip()`을 호출해야 해.

(코드 조각 3: 그림 그리고 화면에 보여주기!)

```
# ... (색깔 정의, 창 생성 코드는 그대로) ...

# 1. 배경색을 파란색으로 채우자! (아직 스케치북에만)
window.fill(BLUE)

# 2. 이제 그림을 그려볼까?
# 선 그리기: window(어디에), RED(무슨색으로), (0,0)(시작점), (100,200)(끝점), 5(굵기)
pygame.draw.line(window, RED, (0, 0), (100, 200), 5)

# 원 그리기: window(어디에), WHITE(무슨색으로), (중심점X, 중심점Y), 100(반지름), 5(굵기)
# 여기서 WINDOW_WIDTH//2는 창 가로 크기의 딱 절반! //는 나누고 소수점 버리는 연산자.
pygame.draw.circle(window, WHITE, (WINDOW_WIDTH//2, WINDOW_HEIGHT//2), 100, 5)
# 굵기가 0이면? 속이 꽉 찬 원이 그려져!
```

```
pygame.draw.circle(window, YELLOW, (WINDOW_WIDTH//2, WINDOW_HEIGHT//2 - 50), 50, 0) # 노란색 꽉찬 원, 살

# 사각형 그리기: window(어디에), CYAN(무슨색으로), (왼쪽위X, 왼쪽위Y, 가로길이, 세로길이)
pygame.draw.rect(window, CYAN, (500, 10, 80, 100)) # x=500, y=10 위치에 80x100 크기
pygame.draw.rect(window, MAGNETA, (400, 200, 50, 150)) # 다른 위치, 다른 크기!

# The main game loop
running = True
while running:
  for event in pygame.event.get():
    if event.type == pygame.QUIT:
      running = False

  # 3. 자, 지금까지 스케치북에 그린 모든 것을 화면에 "짠!" 하고 보여주자!
  # 이 줄이 없으면 아무것도 안보여! (혹은 한번만 보이고 안바뀔수도)
  pygame.display.update()

#End the game
pygame.quit()
```

슬기: 와!!! `pygame.display.update()`를 게임 루프 안에 넣으니까 다 보여요! 파란 배경에 빨간 선, 하얀 테두리 원, 노란색 꽉 찬 원, 하늘색 사각형, 자홍색 사각형까지! 신기하다!

지혜: `pygame.draw.line()`은 선 그리는 거, `pygame.draw.circle()`은 원 그리는 거, `pygame.draw.rect()`는 사각형 그리는 거네요! 파라미터가 좀 많은데…

코드쌤: 맞아, 지혜야. 처음엔 좀 많아 보일 수 있지만, 하나씩 뜯어보면 간단해.

- `pygame.draw.line(surface, color, start_pos, end_pos, width)`:
  - `surface`: 그림 그릴 도화지 (우리 `window` 창!)
  - `color`: 아까 정한 색깔 (예: `RED`)
  - `start_pos`: 선 시작점 좌표 `(x, y)` (예: `(0, 0)`은 왼쪽 맨 위)
  - `end_pos`: 선 끝점 좌표 `(x, y)` (예: `(100, 200)`)
  - `width`: 선 굵기 (숫자가 클수록 굵어져!)
- `pygame.draw.circle(surface, color, center, radius, width=0)`:
  - `center`: 원 중심 좌표 `(x, y)`
  - `radius`: 원 반지름
  - `width`: 테두리 굵기. **이게 0이면 속이 꽉 찬 원이 그려져!** 0보다 크면 그 숫자만큼의 굵기를 가진 테두리만 그려지고.
- `pygame.draw.rect(surface, color, rect)`:

- `rect`: 사각형 정보인데, (왼쪽 위 x, 왼쪽 위 y, 가로 길이, 세로 길이) 이렇게 4개 숫자를 묶어서 전달해.

슬기: 아하! `pygame.circle`에서 마지막 숫자가 `0`이면 색칠된 원, `0`이 아니면 테두리만 있는 원이 되는 거군요! `WINDOW_WIDTH//2`는 창 가로 길이의 중간이라는 뜻이네요! 똑똑한데요?

지혜: 그럼 `pygame.display.update()`는 게임 루프 안에서 계속 호출되어야 하는 건가요? 만약 그림이 바뀌지 않으면 한 번만 호출해도 되지 않나요?

코드쌤: 아주 좋은 질문이야, 지혜! 만약 우리가 그린 그림이 게임 내내 전혀 바뀌지 않는다면, 그림 그리는 코드를 루프 바깥에 두고, `pygame.display.update()`도 루프 바깥에서 (그림 그린 직후에) 한 번만 호출해도 괜찮아. 하지만 지금 코드처럼 `pygame.display.update()`를 게임 루프 안에 두면, 나중에 우리가 캐릭터를 움직이거나 점수가 바뀌는 것처럼 화면 내용이 계속 변할 때, 그 변화를 매번 즉시 화면에 반영해 줄 수 있어. 그래서 보통은 게임 루프 안에 `update()`를 넣는단다. 지금은 그림이 바뀌진 않지만, 좋은 습관을 들이는 거지!

슬기: 우와! 이제 색깔도 쓰고, 도형도 그리고! 진짜 게임 만드는 것 같아요! 선생님, 그럼 저 도형들 움직이게도 할 수 있어요? 막 날아다니게요!

코드쌤: 물론이지, 슬기야! 그건 우리의 다음 모험이 될 거야! 오늘 배운 그림 그리기를 바탕으로, 다음 시간에는 도형들을 살아서 움직이게 만드는 마법을 배워볼 거란다! 기대해도 좋아!

지혜/슬기: 네! 선생님 최고! 다음 시간 완전 기대돼요! 🚀

코드쌤: 하하, 너희들 덕분에 선생님도 힘이 난다! 오늘 배운 내용으로 여러 가지 색깔과 도형을 마음껏 그려보렴! 창의력을 발휘해서 멋진 작품을 만들어봐!