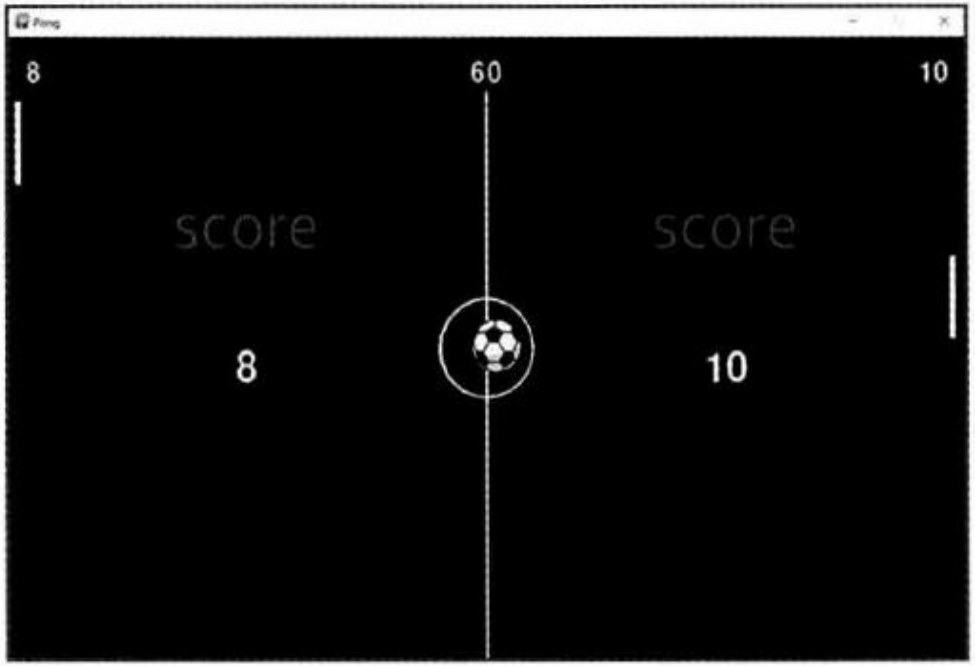
Chapter 17

새로운 게임 오버 스크린

1. 새로운 게임 오버 스크린

- 이제 게임 오버 스크린을 추가합니다.
- 이번에는 이미지를 만들고 그 위에 접수를 전송하는 게 아니라, 스크린에 직접 단어와 점수를 쓰겠습니다.
- 이렇게 하면 이미지 표시에 메모리를 적게 쓸 수 있습니다.
- 이런 게임에서는 메모리를 절약하는 것이 별로 중요하지 않지만, 다른 게임에서는 중요할 수도 있습니다.
- 스크린 완성본은 아래와 같습니다.



1. 새로운 게임 오버 스크린

- "점수(score)"라는 글자를 분홍색으로 두 번 쓰고, 글자 밑에 최종 점수를 씁니다
- 물론 마음대로 화려하게 바꿔도 됩니다.
- 우선 2개의 새 폰트를 만들어야 합니다.
- 다음 코드를 프로그램의 셋업에 씁니다.
- 첫 번째 폰트를 만든 곳이지요.

```
font2 = pygame.font.SysFont("corbel",70)
font3 = pygame.font.Font(None,60)
```

- 두 번째 폰트 (font2)는 corbel이라는 시스템 폰트를 사용하고 크기는 70으로 정합니다.
- 세 번째 폰트 (font3)는 기본 폰트입니다. 크기만 60으로 바꾸었습니다.
- 이런 변수들은 게임 루프 안에서 사용되지만 매 게임 루프마다 바뀌는 것은 아니므로, 셋업에서 만듭니다.

2. 게임 끝내기 1: 먼저 10점 따면 이겨

- 게임 루프 안에 다음 코드를 넣습니다.

```
while 1:
(중략)
    if rscore > 9 or lscore > 9:
        txt = font2.render("score",True,(255,0,255))
        screen.blit(txt,(screen.get_width()/4 - txt.get_width()/2,screen.get_height()
/4))
        screen.blit(txt,(screen.get_width()*3/4 - txt.get_width()/2,screen.get_height()
/4))
        txt = font3.render(str(lscore),True,(255,255,255))
        screen.blit(txt,(screen.get_width()/4 - txt.get_width()/2,screen.get_height()
/2))
        txt = font3.render(str(rscore),True,(255,255,255))
        screen.blit(txt,(screen.get_width()*3/4 - txt.get_width()/2,screen.get_height()
/2))
        while 1:
            for event in pygame.event.get():
                if event.type == QUIT:
                    sys.exit()
            pygame.display.update()

    pygame.display.update()
```

실행

2. “score” 글자 쓰기

- 앞쪽 코드를 설명하겠습니다.
- 이 코드들은 게임 오버 스크린을 불러오기 전에 해야 하는 일입니다.
- 먼저 게임을 끝내는 장치를 만듭니다.

```
if rscore > 9 or lscore > 9:
```

- 먼저 10점을 얻는 사람이 이기도록 하겠습니다.
- 위 코드는 플레이어 둘 중 하나가 9점을 초과하면 참값을 돌려줍니다.
- 따라서 둘 중 하나의 점수가 10이 되면 다음 코드가 실행됩니다.

```
txt = font2.render("score",True,(255,0,255))
```

- 여기서 우리가 만든 두 번째 폰트를 사용하여 “점수(score)"라는 글자를 표시하고 색을 정합니다.
- “점수”는 문자열입니다. 점수(score) 변수와는 전혀 상관없습니다.
- 스크린 위에 점수라는 글자를 표시하는 것뿐입니다.

2. “score” 글자 쓰기

- 여기서도 txt 변수를 쓰네요. 계속 다시 정의하니까 괜찮습니다.

```
screen.blit(txt,(screen.get_width()/4 - txt.get_width()/2, screen.get_height()/4))
screen.blit(txt,(screen.get_width()*3/4 -txt.get_width()/2, screen.get_height()/4))
```

- screen.blit() 함수는 2개의 인자를 갖습니다.
- 첫 번째 인자는 전송할 객체입니다.
- txt 변수 안에 들어 있는 “score" 라는 글자이지요.
- 두 번째 인자는 전송될 위치입니다.
- 1번째 줄에서 x좌표는 screen.get_width()/4 - txt.get_width()/2입니다.
  - 즉, txt를 쓰는 지점의 x좌표는 스크린의 가로 1/4 지점 이라는 뜻입니다.
- 2번째 줄에서 txt를 쓰는 지점의 x좌표는 스크린의 가로 3/4 지점이라는 것을 알 수 있습니다.
- y좌표는 둘 다 스크린의 가장 위에서 1/4만큼 내려온 지점입니다.
- 사실 txt의 높이의 반만큼을 빼지 않았기 때문에, 점수라는 글자의 가운데가 아니라 가장 위가 스크린의 세로 1/4지점이지요.

3. 플레이어별 점수 표시하기


```
txt = font3.render(str(lscore),True,(255,255,255))
screen.blit(txt,(screen.get_width()/4 - txt.get_width()/2,screen.get_height()/2))
txt = font3.render(str(rscore),True,(255,255,255))
screen.blit(txt,(screen.get_width()*3/4 - txt.get_width()/2,screen.get_height()/2))
```

- 위 코드는 세 번째 폰트를 사용하여 lscore와 rscore 변수를 표시합니다.
- 앞쪽에서 한 것과 거의 같아요.

```
while 1:
    for event in pygame.event.get():
        if event.type == QUIT:
            sys.exit()
    pygame.display.update()
```

- 이제 루프 안에 갇혔지요.
- 지금은 게임에서 이기면 게임 오버 스크린이 표시되고 공과 배트가 멈추게 됩니다.

4. 게임 끝내기2: 60초까지 카운트다운

- 10점을 먼저 얻는 사람이 이긴 것으로 정할 수도 있고, 게임 시간을 제한해도 됩니다.
- 시간 제한은 쉽습니다.
- 프로그램의 1번째 줄에 time 모듈을 가져왔겠지요?
- match_start라는 변수를 만들어서 게임 시작부터 흐른 시간을 저장합니다.
- 다음 코드를 프로그램 셋업에 씁니다.

```
match_start = time.time()
```

- 게임이 끝나는 조건도 바꿉니다.

4. 게임 끝내기2: 60초까지 카운트다운

- 게임이 끝나는 조건도 바꿉니다.

```
while 1:
(중략)
    if rscore> 9 or lscore > 9:
    if time.time() - match_start > 60:
```

- 60초가 지나면 위 코드는 참이 되어 게임은 끝납니다.
- 왜냐하면 게임이 시작될 때 match_start가 time.time()과 같은 값으로 설정돼 있기 때문입니다.
- time.time()은 매초 1씩 증가하므로, 60초 뒤에는 time.time()은 match_start보다 60만큼 커지게 돼 참이 됩니다.

5. 스크린 위의 시계

- 스크린 위에 시계를 추가할 수도 있습니다.
- 게임 루프 안에서 배경 선과 원을 그리는 코드 다음에 쓰면 됩니다.

```
while 1:
(중략)
    pygame.draw.line(screen,(255,255,255),(screen.get_width()/2,0),(screen.get_width()
/2,screen.get_height()),3)
    pygame.draw.circle(screen,(255,255,255),(int(screen.get_width()/2),int(screen.get_
height()/2)),50,3)
    txt = font.render(str(int(time.time() - match_start)),True,(255,255,255))
    screen.blit(txt,(screen.get_width()/2 - txt.get_width()/2,20))
```

실행

- 첫 줄부터 봅시다. time.time() - match_start는 프로그램 시작 후 흐른 시간을 알려 줍니다.

5. 스크린 위의 시계

- 첫 줄부터 봅시다. time.time() - match_start는 프로그램 시작 후 흐른 시간을 알려 줍니다.
- 그것을 int() 함수를 통해 정수로 만들고, 그 정수를 str() 함수를 사용해 문자열로 만듭니다.
- 이 문자열을 앞에서 만든 폰트로 표시합니다.
- 이 전체를 txt라는 변수에 저장하는 거지요.
- 2번째 줄은 스크린에 txt를 표시하는 코드입니다.
- 스크린의 중심에서 x축으로는 txt의 너비 반만큼 왼쪽으로 옮기고, y축 방향으로 20픽셀만큼 내립니다.
- 시간을 증가시키는 대신 남은 시간을 표시하려면 아래처럼 바꾸면 됩니다.

```
txt = font.render(str(int(time.time()- match_start 60-(time.time() - match_start))),
True,(255,255,255))
```

실행

Thank You