Chapter 15

골 넣기

1. 퐁 게임에서 골 넣기

- 먼저 각 플레이어의 점수 저장 변수를 2개 만듭니다.
- 다음 코드를 프로그램의 셋업에 쓰세요.

```
rscore = 0
lscore = 0
```

- 공이 스크린에서 반대편 밖으로 나가면 점수를 얻습니다.
- 그러면 플레이어의 점수에 1을 추가하고, 공을 가운데로 옮긴 후 다시 시작시켜야 합니다.

1. 퐁 게임에서 골 넣기

- 아래 음영 표시된 부분을 게임 루프 안, for 루프 밑에 넣으세요.
- 들여쓰기는 1번 합니다.

```
    for bat in bats:
        bat.move()
        bat.draw()

    if ball.x < -50:
        ball = Ball()
        rscore += 1

    if ball.x > 1000:
        ball = Ball()
        lscore += 1
```

1. 퐁 게임에서 골 넣기

- 만약 공이 스크린 왼쪽 끝으로 가면 (ball.x가 -50보다 작으면) 파이썬은 ball=Ball()을 실행시킵니다.
- 이 방법은 이전 게임에서 새로운 악당을 만들 때도 썼지요.
- 이 코드는 공 클래스 (Ball()) 에 인스턴스를 하나 만들고 그 클래스 안의 __init__() 함수를 실행시킵니다.
- 공 하나를 만드는 셈이지요.
- 공은 하나만 있어야 하므로, 공 변수에 새로운 공의 데이터가 들어가고, 낡은 공(또는 낡은 공의 데이터)은 파이썬의 쓰레기 청소 시스템이 치워버립니다.
  - 파이썬에는 실제로 쓰레기 하치장이 있답니다.
- 이렇게 하고 나서 rscore에 1을 더합니다.
- rscore와 lscore는 프로그램의 셋업에서 만듭니다.
- 왜냐하면 공 클래스처럼 이것들도 게임 루프안에서 사용되기 때문입니다.
- 게임 루프 안에서 만들면 매 루프마다 0으로 리셋되기 때문에 게임루프 안에서는 만들 수 없습니다.

2. 점수 표시

- 이제 스크린 위에 점수들을 표시합시다.
- 점수를 표시하려면 폰트 변수를 만들어야지요.

```
lscore = 0
font = pygame.font.Font(None,40)
```

- 점수를 스크린에 표시하는 코드는 게임 루프 안, pygame.display.update 위에 추가합니다.

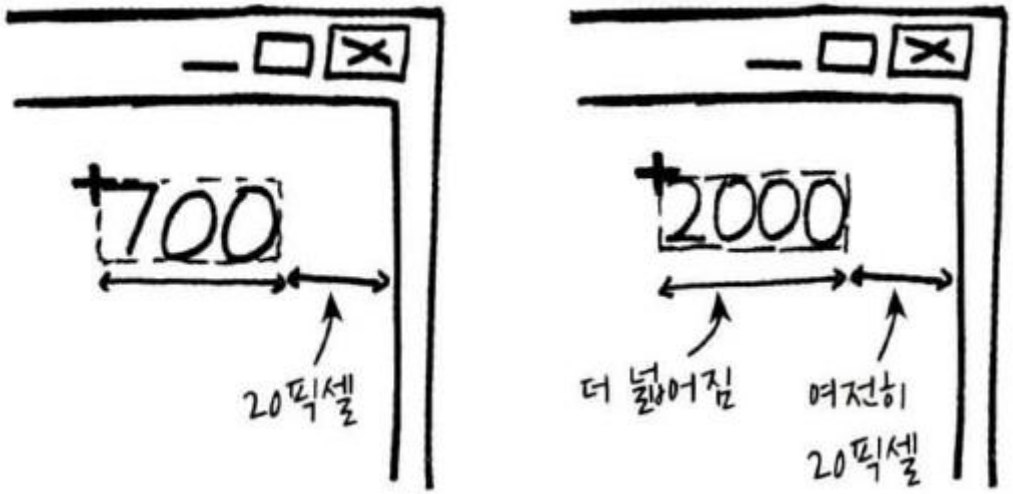
```
txt = font.render(str(lscore),True,(255,255,255))
screen.blit(txt,(20,20))
txt = font.render(str(rscore),True,(255,255,255))
screen.blit(txt,(980-txt.get_width(),20))

pygame.display.update()
```

2. 점수 표시

- 우주 침략자 게임의 점수 표시 코드와는 조금 다르지요?
- 자세히 보면 기능은 같답니다.
- 이 코드는 txt라는 변수를 만들고 관련 속성(attributes)을 정하지요.
  - 3번째 줄에서 또 txt를 써도 상관없습니다.
  - 3번째 줄에 오면 1번째 줄의 코드는 모두 실행 완료된 상태기 때문입니다.
- screen.blit() 함수로 txt를 특정 좌표에 표시합니다.
- 우주 침략자 게임에서는 screen.blit() 함수에 바로 속성(좌표, 크기 등)을 썼습니다.
- 어떻게 하든 상관없지만, 여기서 이렇게 하는 데는 이유가 있습니다.
- 마지막 줄을 보면, screen.blit() 함수 안에서 x좌표를 980-txt.get_width()로 정의했습니다.
- get_width() 함수는 어떤 것의 너비를 가져오는 함수입니다.
- get_width() 함수를 쓰는 이유는 스크린 상에서 깔끔해 보이기 위해서입니다.

2. 점수 표시

- 오른쪽 점수 (txt 변수에 저장된 rscore)의 오른쪽 가장자리가 스크린의 끝에서 20픽셀만큼 멀어지게 하고 싶습니다.
- 왼쪽 점수에서는 쉽지만, 오른쪽 점수에서는 문제가 좀 있습니다.
- 좌표는 가장 위 왼쪽을 기준으로 하기 때문입니다.
- 따라서 x좌표는 980픽셀(스크린의 너비에서 20픽셀만큼 뺀 값)에서 점수의 너비만큼 뺀 값이 됩니다.



2. 점수 표시

- 점수가 증가함에 따라 점수의 너비가 바뀝니다.
- 8은 1보다 넓습니다.1000은 100보다 넓지요.
- get_width() 함수를 쓰면 이런 문제를 해결할 수 있습니다.
- 이 함수는 항상 점수의 너비를 측정하므로, 그 값을 점수의 위치를 정하는 데 사용합니다.
- get_width() 함수는 다른 데에서도 쓸 수 있습니다
- 예를 들어, 우주 침략자 게임에서는 미사일을 미사일들 목록에 더할 때 다음 코드를 사용했습니다

```
missiles.append(Missile(self.x+50))
```

2. 점수 표시

- (self.x+50)는 파이터의 한가운데의 위치입니다
- 파이터의 너비가 100픽셀이라는 것은 알고 있습니다
- 너비를 모르거나, 파이터들의 너비가 모두 다르면, 코드를 다음과 같이 써도 됩니다.

```
missiles.append(Missile(self.x+fighter_image.get_width()/2))
```

- 비슷한 방법으로, 파이터의 y좌표를 정할 때는 다음과 같이 썼습니다

```
screen.blit(fighter_image, (self.x, 591))
```

- 하지만 591이라는 숫자는 스크린의 높이에서 파이터의 높이를 직접 빼어 구한 것입니다. 이런 일은 파이썬한테 시켜도 됩니다

```
screen.blit(fighter_image, (self.x, screen.get_height()-fighter_image.get_height()))
```

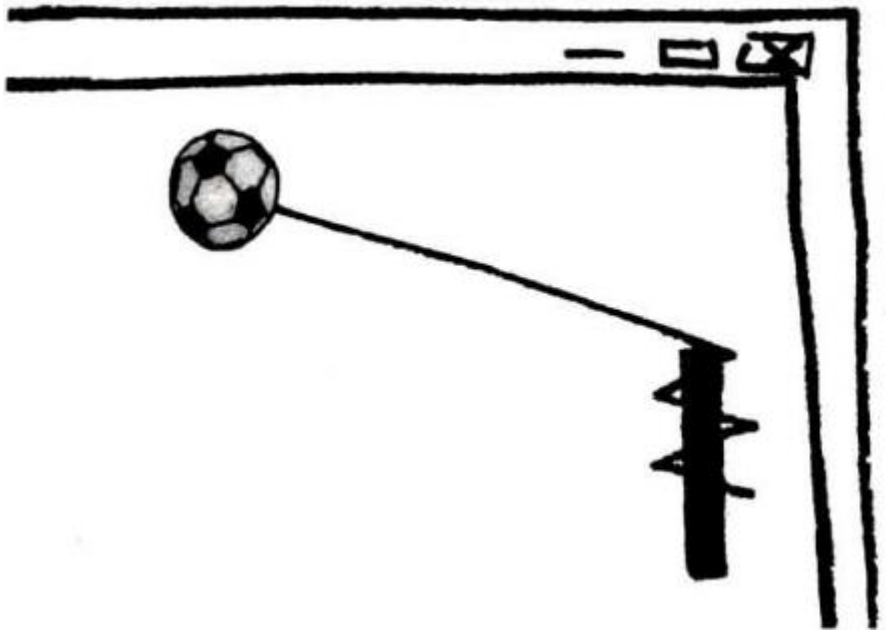
2. 점수 표시

- 다시 돌아와서, 퐁 게임에서는 오른쪽 점수(rscore)를 스크린에 표시하기 위해 다음과 같은 코드를 사용해도 됩니다
- 앞쪽과 다른 점은, 스크린의 너비도 숫자를 직접 쓰지 않고 get_width() 함수로 가져왔다는 점입니다

```
screen.blit(txt, (screen.get_width()-20-txt.get_width(), 20))
```

3. 버그 수정: 공이 배트에 걸린 경우

- 공이 배트에 맞으면 튕겨 나가야 하는데 가끔 여전히 배트에 닿은 채로 있을 때가 있습니다.
- 그러면 다음 게임 루프에서는 튕기기 함수(bounce())가 실행돼 공을 튕깁니다.
- 이 버그를 발견하지 못했다면, 배트가 움직일 때 배트의 가장 끝으로 공을 쳐보세요.
- 공이 배트에 잡힐 때가 있을 겁니다. 그림으로 그리면 다음과 같습니다.



3. 버그 수정: 공이 배트에 걸린 경우

- 이 버그는 여러 방법으로 수정할 수 있습니다.
- 여기서는 공이 배트 쪽으로 움직일 때만 튕기도록 만들겠습니다.
- 공이 1번 튕겨서 배트 반대쪽으로 날아가지만 여전히 배트와 조금 겹친 상태일 때는 튕기지 않도록 하면 됩니다.
- 그러려면 공이 움직이는 방향을 확인해야 합니다.
- dx가 양수면 공은 오른쪽으로 움직입니다.
- dx가 음수면 왼쪽으로 움직이고요.
- 오른쪽으로 공이 움직이는 상태에서, 공의 오른쪽이 배트와 닿으면 튕겨야 합니다.
- 왼쪽으로 공이 움직이는 상태에서는 공의 왼쪽이 배트와 닿으면 튕겨야 하고요.
- 이 외의 경우, 공은 계속 움직여야 합니다.

3. 버그 수정: 공이 배트에 걸린 경우

- 이렇게 만들려면 배트에 왼쪽과 오른쪽을 표시해야 합니다.
- 1은 오른쪽이고 1은 왼쪽입니다.
- 먼저 배트의 __init__() 함수에 사이드 (side)라는 인자를 추가합니다.

```
class Bat:
    def __init__(self,ctrls,x,side):
```

- 이제 __init__() 함수는 (self를 제외한) 3개의 인자를 요구합니다.
- 그러면 배트들 리스트를 수정해 인자를 알려 주어야 합니다.

```
bats = [Bat([K_a,K_z],10,-1), Bat([K_UP,K_DOWN],984,1)]
```

3. 버그 수정: 공이 배트에 걸린 경우

- 왼쪽 배트의 x좌표는 10이고, 사이드 변수의 값은 1로 주었습니다.
- 오른쪽 배트의 x좌표는 984이고, 사이드 변수의 값은 1로 주었습니다.
- 이제 배트 클래스의 __init__() 함수를 업데이트해 사이드 변수의 값이 전달되도록 합니다.

```
def__init__(self,ctrls,x,side):
    self.ctrls = ctrls
    self.x = x
    self.y = 260
    self.side = side
```

- 공의 튕기기 함수에 조건을 추가합니다.

```
def bounce(self):
    if self.y <= 0 or self.y >= 550:
        self.dy *= -1
    for bat in bats:
        if pygame.Rect(bat.x,bat.y,6,80).colliderect(self.x,self.y,50,50) and abs(self.
dx)/self.dx == bat.side:
            self.dx *= -1
```

실행

4. 절댓값

- abs() 함수는 어떤 수의 절댓값을 구합니다.
- 절댓값은 0을 제외한 모든 수를 양수로 만들지요.
- 예를 들어 -6의 절댓값은 6입니다. 6의 절댓값은 그대로 6이지요.
- 어떤 수의 절댓값을 해당 수로 나누면, 그 결과는 -1 또는 1이 됩니다.
- 예를 들어, -6의 절댓값을 -6으로 나누면 6/-6이므로 -1이 됩니다.
- 9의 절댓값을 9로 나누면 9/9이므로 1이 됩니다.
- abs(self.dx)/self.dx는 공에게 1 또는 -1이라는 방향을 줍니다.
- dx가 음수이면 -1이 됩니다. dx가 양수이면 1이 됩니다.
- 오른쪽에 있는, x좌표가 984인, 배트의 사이드 변수의 값은 1입니다.
- 왼쪽에 있는 배트의 사이드 변수의 값은 -1입니다.
- 따라서 abs(self.dx)/self.dx == bat.side는 공의 방향이 배트의 사이드 변수의 값과 같은지를 확인합니다.
- 같을 때만 dx에 -1을 곱하니까 같을 때만 공이 튕겨져 나가게 됩니다.

5. 배경 그리기

- 배경에 중심선과 중심원을 추가합니다.
- screen.fill() 함수 바로 밑에 아래 코드를 씁시다.
- 스크린을 채운 다음에 선과 원을 그려야 하니까요.

```
screen.fill((0,0,0))

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)
```

실행

- 앞에서 배웠듯이 line() 함수와 circle() 함수 둘 다 5개의 인자를 가집니다
  - 두께를 포함하면 6개.
- 이번에는 숫자를 직접 써 넣지 않고 get_width() 함수와 get_height() 함수를 인자로 주었습니다.
- 이렇게 하면 스크린의 크기를 바꿨을 때 스크린 위 객체의 위치가 자동으로 수정된다는 장점이 있습니다.

5. 배경 그리기

- line() 함수의 세 번째, 네 번째 인자를 봅시다.
- 세 번째 인자는 직선이 시작하는 점의 좌표입니다.
- x좌표는 screen.get_width()/2 이므로 스크린의 너비의 한가운데 입니다.
- y좌표는 0입니다.
- 네 번째 인자는 직선이 끝나는 점의 좌표입니다.
- x좌표는 세 번째 인자와 동일하지만 y좌표는 스크린의 높이입니다.
- 따라서 스크린 크기에 상관없이 스크린 한가운데를 수직으로 내려오는 직선이 그려집니다.
- circle() 함수로 원을 그릴 때는 가장 위 왼쪽이 아니라 원의 중심의 좌표를 씁니다.

Thank You