Zestaw 5

Wstęp

Nasza znajomość Pythona jest wystarczająca, aby zacząć pisać prostą grę. W tym celu zapoznamy się z podstawowymi funkcjami biblioteki („silnika" do tworzenia gier) pygame. Nie będzie tu kompletnego „samouczka", tylko minimalne, konieczne do napisania zadań, wprowadzenie. Ciekawą instrukcję (znalezioną „w internecie") załączam jako jeden z materiałów w Pegazie (plik Pygame_Python.pdf). Ponadto na stronie https://www.pygame.org dostępne jest sporo przykładowych projektów wraz z kodami źródłowymi. Aby możliwe było korzystanie z pygame, oraz zaktualizowaliśmy menadżer pakietów pip, na przykład tak: `python -m pip install -U pip`, to instalujemy (na ten moment zainstaluje się pygame-2.1.2): `pip install pygame`

Szkielet programu (gry) w pygame wygląda następująco:

```
1.  import pygame, sys
2.  pygame.init()
3.
4.  def main():
5.
6.      size = width, height = 800, 600
7.      screen = pygame.display.set_mode(size)
8.
9.      while True:
10.         for event in pygame.event.get():
11.             if event.type == pygame.QUIT: sys.exit()
12.
13. if __name__ == '__main__':
14.     main()
15.     pygame.quit()
16.     sys.exit()
```

Zaczynając od linii 13. – interpreter Pythona czytając plik źródłowy, definiuje kilka specjalnych zmiennych. Kiedy uruchamiany moduł (plik źródłowy) jest głównym programem, to interpreter przypisze na stałe zakodowany ciąg '__main__' do zmiennej __name__. I jeśli tak jest, to uruchamiamy pożądane funkcje – tutaj main(), a po jej zakończeniu metody zwalniające zasoby (pygame.quit()) oraz wychodzące z Pythona (sys.exit()).

Linia 1. Importujemy pożądane moduły. Linia 2. Inicjalizacja wszystkich modułów pygame (nie wszystkie może potrzebujemy, ale jest to wygodne). Linia 4. i kolejne – definicja naszej funkcji main() – nie musi się tak nazywać, ale dobrze wygląda. Linie 6-7. Definiujemy krotkę (tuplę) o nazwie size, a przy okazji zmienne szerokość i wysokość, są to oczywiście rozmiary naszego okienka, po czym inicjalizujemy je. Linie 9-11. Program pygame działa na zasadzie nieskończonej pętli, w której można pobierać różnego rodzaju zdarzenia. Linie 10-11. są konieczne, żeby okienko dało się zamknąć.

Z powyższym kodem proszę zrobić dwa testy:

1) zamiast linii 11-12. wstawić pass, proszę jednak mieć na uwadze, że program tak uruchomiony da się „ubić" tylko przez zewnętrzną interwencję (np. Menedżer programów w Windows).

2) dopisać pod linią 10, a przed linią 11, print(event) i zobaczyć, co się drukuje: jest to bardzo pouczające, bo zobaczymy, jak pygame reaguje na ruchy myszki, klikanie, wciskanie klawiszy itd. Proszę spróbować!

Dodajmy kilka linii kodu w main():

```
1.  def main():
2.      clock = pygame.time.Clock()
3.
4.      pygame.mixer.music.load(r'C:\jakas_sciezka\music.mp3')
5.      pygame.mixer.music.play(-1)
6.
7.      size = width, height = 800, 600
8.      screen = pygame.display.set_mode(size)
9.
```

```
10.     while True:
11.         for event in pygame.event.get():
12.             if event.type == pygame.QUIT: sys.exit()
13.
14.         clock.tick(60)
15.         print(clock.get_fps())
16.         #pygame.time.delay(50)
```

Linia 2. Pobieramy obiekt zegar, dzięki któremu można ustawić „frame rate" w naszej grze – tu ustawiłem 60 fps w linii 14. Aby się przekonać jak jest naprawdę, w linii 15. drukujemy (na razie do terminalu, nie w okienku) faktyczną liczbę klatek na sekundę. Proszę poeksperymentować z różnymi wartościami! Można celowo ustawić niski „frame rate" i uzyskać efekt „poklatkowy". Czasem dla płynności akcji przydaje się jednak co innego – opóźnienie realizacji pętli, proszę odkomentować linię 16. i też poeksperymentować (delay jest w ms), przy okazji obserwując fps.

Dla urozmaicenia – dodajmy też muzykę. Linia 4. nie wymaga komentarza, proszę oczywiście użyć jakiegoś swojego pliku .mp3 (lub ściągnąć plik dodany do materiałów zadania i wpisać właściwą ścieżkę). Linia 5. uruchamia odtwarzanie (opcja -1 to nieskończona pętla, inne opcje – zajrzeć do dokumentacji na stronie ↳ https://www.pygame.org/docs/ref/music.html#pygame.mixer.music.play)

Dodajmy teraz tytuł, ikonkę oraz tło – przeskalowane i pozycjonowane np. w środku.

```
1.  def main():
2.      clock = pygame.time.Clock()
3.
4.      pygame.display.set_caption('Tytul naszego okienka')
5.      icon = pygame.image.load('jakies_zdjecie.jpg')
6.      pygame.display.set_icon(icon)
7.
8.      pygame.mixer.music.load(r'C:\jakas_sciezka\music.mp3')
9.      pygame.mixer.music.play(-1)
10.
11.     size = width, height = 800, 600
12.     screen = pygame.display.set_mode(size)
13.
14.     image = pygame.image.load(r'C:\jakas_sciezka\moon.jpg')
15.     image = pygame.transform.scale(image, size)
16.
17.     surf_center = (
18.         (width-image.get_width())/2,
19.         (height-image.get_height())/2
20.     )
21.
22.     screen.blit(image, surf_center)
23.     pygame.display.flip()
24.
25.     while True: # reszta jak poprzednio
```

Linie 4-6. sprawią, że nasze okienko będzie miało zdefiniowany przez nas tytuł, a jego rogu pojawi się mała ikonka (miniaturka zdjęcia, które wybierzemy). Linia 14. Wczytanie obrazka tła oraz ewentualne przeskalowanie go (linia 15.) do wielkości naszego okienka. Linie 17-20. to krotka wyliczająca środek. Parametr ten użyty jest w linii 22. Linia 22-23. Narysowanie tła (drugi parametr to punkt lewy górny okna), zaś flip() odświeża (przerysowuje) wszystko.

A teraz tuż przed linią 23. dodajmy jeszcze obrazek np. z piłeczką. Proszę jakąś samemu narysować, albo znaleźć, warto jako .gif z przezroczystą warstwą dookoła piłki!

```
1.      ball = pygame.image.load('ball.gif')
2.      screen.blit(ball, (width/2, height/2))
3.      ballrect = ball.get_rect(center=(width / 2, height / 2))
4.      pygame.display.flip()
```

Linia 3. to odczytanie prostokątnych wymiarów obrazka, co przyda się do sprawdzania kolizji obiektu np. z krawędziami okna. Można się postarać i wypozycjonować piłkę w samym środku.

Dopiszę jeszcze teraz pewne zmienne potrzebne do mechaniki naszej piłki – prędkość speed (początkową, składowe x i y zero), „przyspieszenie" (accel, również jako składowe). Cały kod main() do pętli while może wyglądać mniej więcej tak:

```
1.  def main():
2.      clock = pygame.time.Clock()
3.
4.      pygame.display.set_caption('Tytul naszego okienka')
5.      icon = pygame.image.load('ikonka.jpg')
6.      pygame.display.set_icon(icon)
7.
8.      pygame.mixer.music.load(r'C:\jakas_sciezka\music.mp3')
9.      pygame.mixer.music.play(-1)
10.
11.     size = width, height = 800, 600
12.     screen = pygame.display.set_mode(size)
13.
14.     speed = [0, 0]
15.     accel = [0.1, 0.1]
16.
17.       image=pygame.image.load(r'C:\jakas_sciezka\moon.jpg')
18.     image = pygame.transform.scale(image, size)
19.
20.     surf_center = (
21.         (width-image.get_width())/2,
22.         (height-image.get_height())/2
23.     )
24.
25.     screen.blit(image, surf_center)
26.     ball = pygame.image.load('ball.gif')
27.     ball = pygame.transform.scale(ball, (ball.get_width()//2, ball.get_height()//2))
28.
29.     screen.blit(ball, (width/2, height/2))
30.
31.     ballrect = ball.get_rect(center=(width/2, height/2))
32.     pygame.display.flip()
33.
34.
35.     while True:
```

Teraz jak odczytujemy i działamy w pętli.

```
1.      while True:
2.          clock.tick(60)
3.          pygame.time.delay(50)
4.
5.          for event in pygame.event.get():
6.              if event.type == pygame.QUIT: sys.exit()
7.
8.          keys = pygame.key.get_pressed()
9.          if keys[pygame.K_ESCAPE]: sys.exit()
10.
11.         if keys[pygame.K_UP]:
12.             pass # zamienić na jakieś przeliczenie
13.         elif keys[pygame.K_DOWN]:
14.             pass
15.         elif keys[pygame.K_LEFT]:
16.             pass
17.         elif keys[pygame.K_RIGHT]:
18.             pass
19.
20.         ballrect = ballrect.move(speed)
21.         if ballrect.left < 0 or ballrect.right > width:
22.             speed[0] = -speed[0]
23.         if ballrect.top < 0 or ballrect.bottom > height:
24.             speed[1] = -speed[1]
25.
26.         screen.blit(image,surf_center)
27.         screen.blit(ball,ballrect)
28.         pygame.display.flip()
```

Wciśnięcie klawiszy odczytujemy poprzez key.get_pressed(), otrzymujemy wartości logiczne, wpisane to tablicy keys. I jak widać, linia 9., można np. zareagować na fakt wciśnięcia Esc i wyjść z programu. Analogicznie można reagować na strzałkę w górę, dół, lewo, prawo (linie 11-18.). A jak reagować – to będzie właśnie przedmiotem zadania. Ruch piłki to linia 20., zmiana zwrotu odpowiedniej składowej prędkości to linie 21-24. Można na początku oczywiście ustawić jakąś „prędkość” początkową (jak wyżej, speed). Ostatnie trzy linie to przerysowanie i odświeżenie.

Zadania

1. Na bazie kodu jak wyżej, zmodyfikować kod w taki sposób, żeby piłka (ruszała) przyspieszała w kierunku, w którym wciśnięta jest strzałka. Z prostego wzoru na prędkość „w ruchu jednostajnie przyspieszonym" mamy: $v_i = v_{0i} + a_i \cdot t$, gdzie i to składowa (x, y). Oczywiście u nas początkowa prędkość może być (0, 0). Przyspieszenie („umownie") accel ma jakieś wartości (może być 1), natomiast poprzez t czas rozumieć należy coś takiego, że jeśli dana strzałka jest wciąż wciśnięta, to zwiększamy prędkość (w tym sensie t może być równe 1, albo dowolnie inaczej). Efekt końcowy ma być taki: żeby piłka, początkowo nieruchoma, mogła być wprawiana w ruch i sterowana we wszystkich kierunkach za pomocą strzałek. Proszę poeksperymentować! Uwaga: piłka w zadaniu 1 i 2 powinna się odbijać „doskonale sprężyście" od granic aktywnego ekranu gry [**zadanie za 1,5 pkt**]
2. Na bazie powyższego kodu zrobimy symulację ruchu w polu grawitacyjnym. Proszę zatem ustalić jakieś wartości prędkości początkowej piłki, przyspieszenie ma składową pionową (0, 9.81) (składowe x, y). I teraz, jeśli piłka jest nieruchoma na początku – to będzie to spadek swobodny (z przyspieszeniem g = 9.81 m/s^2), jeśli „rzucona w górę" ($v_y > 0$), to rzut pionowy, jeśli „rzucona w bok" ($v_x > 0$) to rzut poziomy i ogólnie – rzut ukośny. Piłka powinna poruszać się realistycznie (w sensie: należy wyliczać jej prędkości według ruchu przyspieszonego w pionie i jednostajnego w poziomie). Oczywiście, podobnie jak w zadaniu 1., wartość przyspieszenia (numerycznie) może być dowolnie dobrana tak, żeby ruch odbywał się płynnie, nie za wolno i nie za szybko. Proszę odbijać piłkę doskonale sprężyście (bez strat energii! – czyli w sumie w nieskończoność) [**zadanie za 1,5 pkt**].
3. Najpierw należy przestudiować załączony kod (main.py wszystko w jednym pliku), jest to klasyczna gra w ping-ponga, napisana z użyciem znanej nam biblioteki pygame. Proszę po kolei przestudiować kod, który jest komentowany i choć (ewentualnie) zawiera rzeczy nowe, to można się domyślić o co chodzi. W szczególności na początku są definicje dwóch klas Rakietka i Pilka, które zapisane są jako dziedziczące z klasy pygame.sprite.Sprite (proszę zobaczyć w kodzie jak to wygląda). Klasy są dość proste, ich metody dbają o zmianę i sprawdzenie położeń granicznych oraz ustalanie (np. losowanie w pewnym zakresie) wartości prędkości piłki. Program zaczyna się od narysowania ekranu, rakietek, piłki (piłka jest o rozmiarze 10x10 punktów), utworzeniu listy widzialnych w grze obiektów (właśnie odziedziczonych z klasy Sprite). Sama mechanika ruchów rakietek powinna być już znana z poprzednich zadań, ciekawa jest metoda collide_mask sprawdzająca czy dane dwa obiekty nie są ze sobą w styczności / kolizji, jeśli tak jest, to na rzecz piłeczki wołamy metodę bounce(), która zmienia (i trochę losuje) składową prędkości piłki po odbiciu. Zadanie: po przestudiowaniu i uruchomieniu kodu zadanie będzie polegać na takim jego zmodyfikowaniu, żeby: (a) rakietka była tylko jedna, poruszająca się w poziomie na dole ekranu (w lewo i prawo, strzałkami), (b) piłeczka uruchamiana losowo z góry, punkty mają być naliczane za poprawne odbicie od rakietki, (c) gra ma się zakończyć jeśli piłeczka minie rakietkę i zderzy się ze ścianą – wtedy powinien się wyświetlić wynik końcowy oraz dotychczasowy najwyższy wynik. Najlepszy wynik zapisywać do i odczytywać z pliku. Oczywiście pionowa linia jest teraz zbędna. Innymi słowy – przerobić to na grę „jednoosobową" [**zadanie za 2 pkt**].