

ZUZOHELL

Czyli gra typu „bullethell”

Autor: Zuzanna Dąbrowa



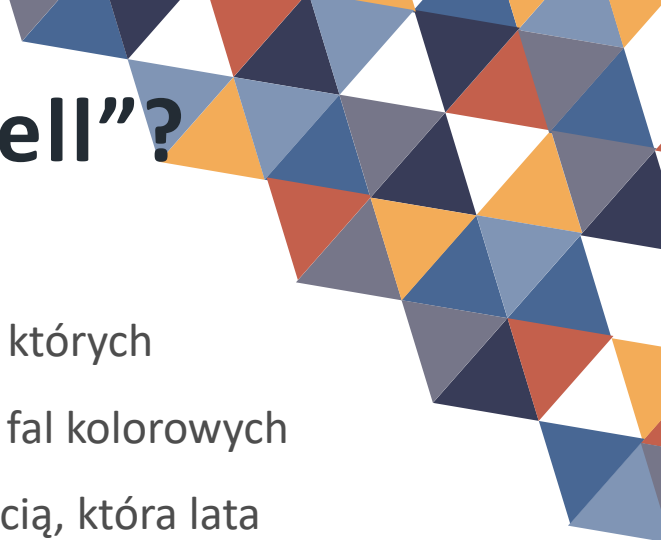


PLAN PREZENTACJI

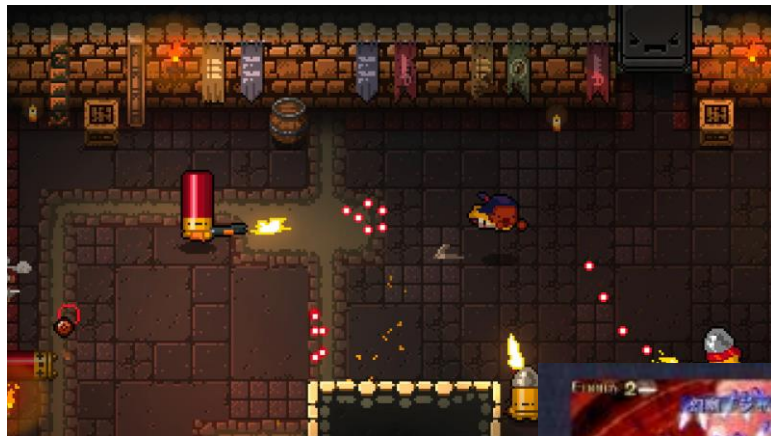
1. Czym jest gra typu „bullethell”?
2. Omówienie projektu i jego struktury
3. Wprowadzenie do biblioteki pygame
4. Treść zadań
5. Źródła i bibliografia

Czym jest gra typu „bullethell”?

Termin „bullethell” odnosi się do podgatunku strzelanek, których głównym celem jest unikanie pozornie niekończących się fal kolorowych kul. W tych grach gracz steruje małym statkiem lub postacią, która lata wokół otoczenia, strzelając do fal wrogów, aż w końcu dotrze etapu z ostatecznym bossem. Prawdziwa strzelanka z bullethell sprawdza umiejętności gracza w rozpoznawaniu wzorów w strumieniu pocisków, a następnie poruszaniu się po ograniczonym polu.



Przykłady popularnych gier bullethell



Enter the Gungeon - Doge Roll



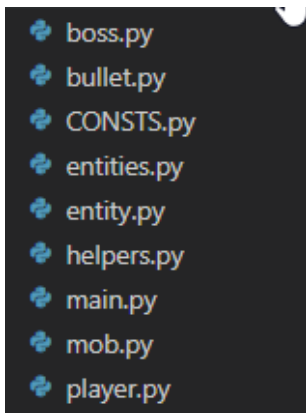
The Binding of Issac - Edmund McMillen



Touhou Eosd – ZUN

Omówienie projektu

Struktura projektu



-main.py

Główny plik. Obsługuje główną pętlę gry, wyświetlanie oraz zarządzanie ścieżkami dźwiękowymi

-CONSTS.py

Jest to plik zawierający listę stałych użytych w projekcie

-helpers.py

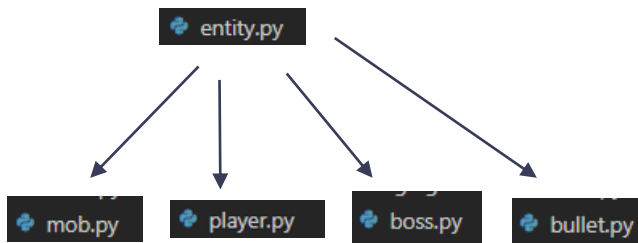
Zawiera funkcje, które obsługują często powtarzalne akcje w głównej pętli programu

-entities.py

Przechowuje zmienne z obiektami znajdującymi się aktualnie w grze

-entity.py

Klasa rodzic. Dziedziczą po niej klasy Mob, Bullet, Player, oraz Boss.



Omówienie biblioteki pygame



Instalacja: `pip install pygame`

- To stworzona przez Pete Shinnorsa biblioteka przeznaczona do tworzenia gier komputerowych oraz aplikacji multimedialnych w języku Python.
- Pygame daje nam do dyspozycji **GUI** (Graficzny interfejs użytkownika,)który umożliwia na wyświetlanie grafiki, odtwarzanie dźwięków, śledzenie czasu, obsługę myszy i joysticka, obsługę CD, czy renderowanie czcionek TTF.
- Pygame jest darmowe, wydany na licencji LGPL
- Można za jego pomocą tworzyć gry open source, freeware, shareware i komercyjne.

Podstawowa pętla gry



```
import pygame
pygame.init()
YELLOW = (255,255,0)
WHITE = (255, 255, 255)
WINDOW_W = 400
WINDOW_H = 600

screen = pygame.display.set_mode([WINDOW_W, WINDOW_H])

running = True

while running:
    screen.fill(WHITE)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    pygame.draw.circle(screen, YELLOW, (WINDOW_W/2, WINDOW_H/2), 30)
    pygame.display.update()

pygame.quit()
```

Przykładowa obsługa eventów

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_UP:
            print("Player moved up!")
        elif event.key == pygame.K_LEFT:
            print("Player moved left!")
        elif event.key == pygame.K_DOWN:
            print("Player moved down!")
        elif event.key == pygame.K_RIGHT:
            print("Player moved right!")
```

Dodatkowa uwaga: W przypadku, jeśli chcemy utrzymać akcję po naciśnięciu klawisza, powinniśmy skorzystać z `keystate = pygame.key.get_pressed()`

W podanym przykładzie po lewej, aby wyświetlić informację o ruchu musimy od nowa wciskać odpowiedni klawisz.



Obsługa muzyki

```
import pygame.mixer

pygame.mixer.init()
EXAMPLE_SOUND = pygame.mixer.Sound(,yourpath/example.mp3')
pygame.mixer.set_num_channels(0)
pygame.mixer.set_reserved(0)
pygame.mixer.Channel(0).play(EXAMPLE_SOUND, -1)
#można też użyć po prostu
#EXAMPLE_SOUND.play()
```

Jak zatrzymać muzykę?

```
if pygame.mixer.Channel(1).get_busy() == True:
    pygame.mixer.Channel(1).stop()
```

Inne funkcje, które mogą się przydać podczas rozwiązywania zadań

```
pygame.display.set_caption("podpis")
icon = pygame.image.load('yourpath/img.png')
pygame.display.set_icon(icon)
```

Wstawienie ikony i podpisu gry na pasku

```
class Player(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.Surface((50, 50))
        self.image.fill((255,0,0))
        self.rect = self.image.get_rect()
        self.rect.center = (WIDTH / 2, HEIGHT / 2)
```

Przykładowe użycie wbudowanej w pygame klasy Sprite . W rezultacie zobaczymy jeżdżący na ekranie kwadrat

```
def update(self):
    self.rect.x += 1
    if self.rect.left <= 0:
        self.rect.right = WIDTH
```

```
all_sprites = pygame.sprite.Group()
player = Player()
all_sprites.add(player)
```



UWAGA! Zadania są ze sobą powiązane!



Zadania

- ▶ 1. Napisz grę, która po uruchomieniu wyświetli kółko, które po przyciśnięciu klawiszy strzałek będzie Poruszać się po całej planszy (kółko nie może wychodzić poza obszar ekranu gry)
- ▶ 2. Napisz grę, która wykryje kolizję dwóch kółek. Możesz do tego użyć `pygame.sprite.Sprite()`, lub obliczyć wektor odległości pomiędzy nimi. Jedno z kółek powinno mieć możliwość sterowania Strzałkami. Kółka powinny zmienić kolor w trakcie Kolizji.
- ▶ 3. Zrób coś kreatywnego! Zamiast kółek spróbuj dodać własne obrazki. Dodaj dźwięk, który odtworzy się RAZ podczas rozpoczęcia kolizji, dodaj własny tytuł gry. Zaprogramuj ruch drugiego kółka.

<https://realpython.com/pygame-a-primer/>

<https://riptutorial.com/pygame/example/18046/event-loop>

<https://stackoverflow.com/questions/29640685/how-do-i-detect-collision-in-pygame>

<https://www.pygame.org/docs/tut/SpriteIntro.html>

<https://www.pygame.org/news>

<https://www.giantbomb.com/bullet-hell/3015-321/>

Źródła

Warto do nich zerknąć podczas

Rozwiązywania zadań

A large, solid orange triangle is centered in the background of the slide.

Dziękuję za uwagę