Peaky Climber

Michał Niezgoda Krystian Kiejno

Agenda

1. **PyGame**
   - Czym jest Pygame?
   - Historia i rozwój
   - Instalacja
   - Podstawowe funkcjonalności
   - Dodawanie dźwięku i muzyki
   - Interakcja pomiędzy obiektami
   - Dodawanie sprite’ów (obrazów)

2. **Wykonanie projektu**
   - Dlaczego podejście obiektowe?

Czym jest 

Pygame to zestaw modułów Pythona przeznaczonych do pisania gier wideo. Pygame dodaje funkcjonalność do biblioteki SDL. Pozwala to na tworzenie w pełni funkcjonalnych gier i programów multimedialnych w języku Python.

SDL biblioteka programistyczna ułatwiająca tworzenie gier oraz programów multimedialnych. Zapewnia niskopoziomowy dostęp do sprzętu audio, klawiatury, myszy itd.

Instalacja

```
pip install pygame
```

Jedną z dużych zalet Pygame jest prostota jego użycia jedyne, co trzeba to zainstalować bibliotekę pygame przy użyciu instalera pakietów **pip** oraz import biblioteki.

```
1    import pygame
```

Podstawowe funckjonalnośći

Tworzenie okna startowego

Funkcja pygame.display.set_mode(size)

```
window = pygame.display.set_mode((MAX_WIDTH, MAX_HEIGHT))
```

Ustawia ikonę okna gry

Funkcja pygame.display.set_icon(icon)

```
pygame.display.set_icon(pygame.image.load("./player/player.png"))
```

Ustawianie tytułu okna gry

Funkcja pygame.display.set_caption(title)

```
pygame.display.set_caption("Peaky Climber")
```

Kopiuje zawartość powierzchni źródłowej **source** window w miejscu określonym przez **dest**.

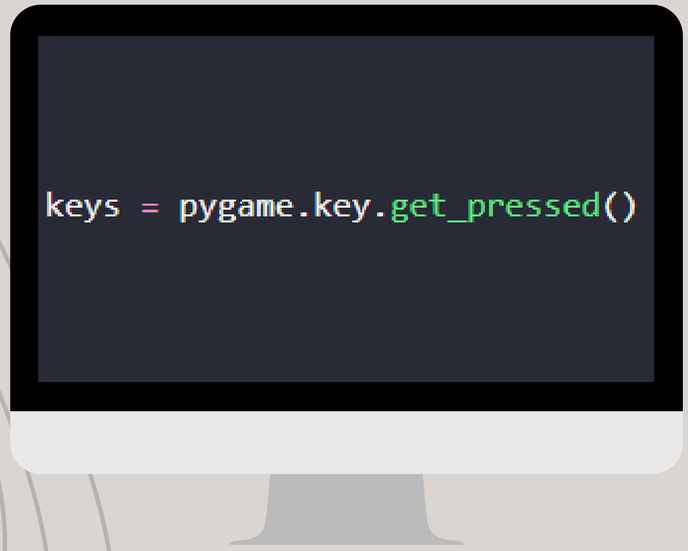
Funkcja window.blit(source, dest)

```
window.blit(menuBackgroundImage,(130,75))
```

Interakcja z użytkownikiem

Biblioteka Pygame dostarcza zaawansowane narzędzia do zarządzania interakcjami, co jest szczególnie przydatne w kontekście tworzenia gier i aplikacji multimedialnych.

Funckja **pygame.key.get_pressed()**



Funkcja **get_pressed()** w Pygame służy do sprawdzania stanu wszystkich klawiszy na klawiaturze. Zwraca ona listę, w której każdy element odpowiada jednemu klawiszowi, a jego wartość wskazuje, czy dany klawisz jest aktualnie wciśnięty czy nie.

Łatwe sterowanie przyciskami

Pygame dostarcza wygodne metody do obsługi klawiszy w grach. Dzięki nim możemy precyzyjnie kontrolować ruch postaci za pomocą strzałek na klawiaturze alfanumerycznej lub odpowiednich klawiszy. Pygame udostępnia stałe, które reprezentują kody klawiszy.

Przykłady przycisków:

- **pygame.K_LEFT**
- **pygame.K_RIGHT**
- **pygame.K_UP**
- **pygame.K_DOWN**

Aktualizacja obiektów na ekranie

Dany obiekt na ekranie posiada swoje koordynaty, aby je zmienić można manipulować zmiennymi rect.x, oraz rect.y, jednak aby zauważyć zmianę na ekranie musimy dodatkowo posłużyć się funkcją **.update()**

```
window.blit(self.image, (self.x , self.y))
self.rect.update(self.x, self.y, 40, 70)
self.y += self.speed
```

Dodawanie dźwięku i muzyki

Pygame oferuje prosty interfejs do odtwarzania dźwięków i muzyki, co umożliwia łatwe dodanie efektów i ścieżek dźwiękowych.

JAK TO DZIAŁA?

- Po wywołaniu funkcji **pygame.mixer.init()**, Pygame przygotowuje moduł mixer do pracy, przygotowując odpowiednie zasoby systemowe.
- Inicjalizacja modułu mixer musi być wykonana przed każdym użyciem funkcji do odtwarzania dźwięków lub muzyki.

Dodawanie dźwięku i muzyki

Wczytywanie plik dźwiękowego z określonej ścieżki i przygotowanie go do odtworzenia.

Funkcja pygame.mixer.music.load()

```
pygame.mixer.music.load("./assets/music.mp3")
```

Rozpoczyna odtwarzanie wcześniej załadowanej muzyki.
Argument -1 wskazuje, że muzyka będzie odtwarzana w nieskończoność.

Funkcja pygame.mixer.music.play()

```
pygame.mixer.music.play(-1)
```

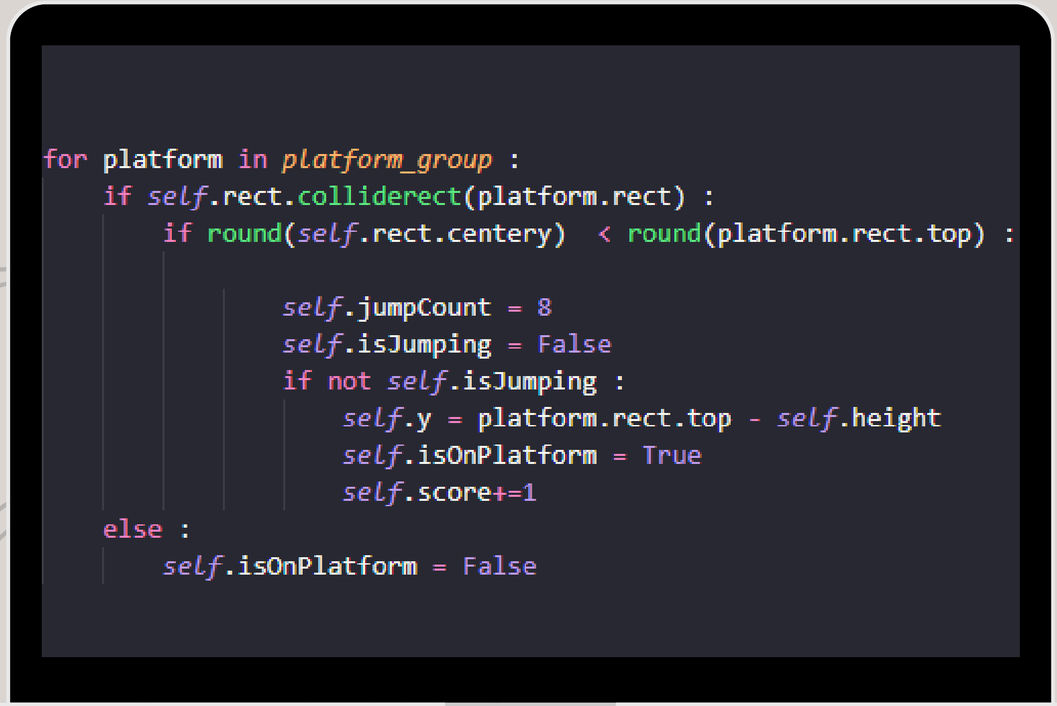
Ustawianie głośności muzyki

Funkcja pygame.mixer.music.set_volume()

```
pygame.mixer.music.set_volume(0.01)
```

Interakcja pomiędzy obiektami

W Pygame interakcja pomiędzy obiektami, takimi jak postaci graczy i elementy otoczenia, jest również wspierana. Poniższa funkcja stanowi podstawowy mechanizm do sprawdzania kolizji między prostokątnymi obiektami w grze.



```
for platform in platform_group :
    if self.rect.colliderect(platform.rect) :
        if round(self.rect.centery)  < round(platform.rect.top) :

                self.jumpCount = 8
                self.isJumping = False
                if not self.isJumping :
                    self.y = platform.rect.top - self.height
                    self.isOnPlatform = True
                    self.score+=1
    else :
        self.isOnPlatform = False
```

Gdy gracz lub inny obiekt gry przemieszcza się, ta pętla sprawdza, czy jego prostokąt kolizyjny **self.rect** nakłada się na prostokąt innego obiektu w grupie platform **platform.rect**. Jeśli tak się dzieje, możemy wywołać określone akcje, np. zwiększyć punktację.

Dodawanie sprite’ów

Grupy przedmiotów są strukturami danych dostarczanymi przez Pygame, które umożliwiają efektywne zarządzanie wieloma sprite'ami jednocześnie. Sprite'y w grze, takie jak postacie graczy, przeciwnicy, elementy otoczenia, itp., są organizowane i zarządzane za pomocą tych grup.

DLACZEGO ICH UŻYWAMY?

Zarządzanie - Ułatwiają zarządzanie wieloma sprite'ami, zapewniając funkcje do dodawania, usuwania i aktualizacji sprite'ów w grupie.
Kolizje - Pozwalają na sprawdzanie kolizji między sprite'ami w grupie, co jest kluczowe dla implementacji mechanik gry, takich jak trafienia gracza, interakcja z innymi obiektami.

Przykład implementacji

Funkcja group.add(sprite)

```
platform = Platform(p_x, p_y, platform_sprite)
platform_group.add(platform)
```

```
fireball = fireballClass(f_x,-100,10)
fireball_group.add(fireball)
```

Dlaczego obiektowo?

Modularyzacja kodu - W podejściu obiektowym możemy łatwo zdefiniować różne klasy obiektów, takie jak Gracz, Platforma, itp. Każda klasa może mieć swoje własne metody i właściwości, co prowadzi do bardziej zorganizowanego i modułowego kodu.

Łatwiejsze zarządzanie obiektami - Obiekty w grze, takie jak Gracz czy Platforma, mogą przechowywać swoje własne stany i zachowania. Na przykład Gracz może przechowywać swoje położenie, prędkość i stan skoku, a Platforma może przechowywać swoje położenie i rozmiar. To ułatwia zarządzanie nimi i aktualizację ich stanu.

Rozszerzalność - Dzięki podejściu obiektowemu możemy łatwo rozszerzać funkcjonalność gry poprzez tworzenie nowych klas obiektów lub dziedziczenie istniejących klas i modyfikowanie ich zachowań. Na przykład możemy łatwo dodać nowe typy przeciwników lub nowe rodzaje platform.

Źródła:

https://www.pygame.org
https://pl.wikipedia.org/wiki/Pygame

Dziękujemy za uwagę