c0b220698f /
ProjExD_05

Code Issues Pull requests Actions Projects Wiki Security Insights Set

C0B22069/score

ProjExD_05 / dinosaur.py

Go to file t

c0b220698f スコア実装完了 41 minutes ago

161 lines (131 loc) · 4.27 KB

Code Blame Raw

```python
import sys
import os
import random

import pygame as pg


SCREEN_HEIGHT = 600
SCREEN_WIDTH = 1100
SCREEN = pg.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))

RUNNING = [
    pg.image.load(os.path.join("ex05/Assets/Dino", "DinoRun1.png")),
    pg.image.load(os.path.join("ex05/Assets/Dino", "DinoRun2.png")),
]
JUMPING = pg.image.load(os.path.join("ex05/Assets/Dino", "DinoJump.png"))
DUCKING = [
    pg.image.load(os.path.join("ex05/Assets/Dino", "DinoDuck1.png")),
    pg.image.load(os.path.join("ex05/Assets/Dino", "DinoDuck2.png")),
]


BG = pg.image.load(os.path.join("ex05/Assets/Other", "Track.png"))


class Dinosaur:
    X_POS = 80
    Y_POS = 310
    Y_POS_DUCK = 340
    JUMP_VEL = 8.5

    def __init__(self):
        self.duck_img = DUCKING
        self.run_img = RUNNING
        self.jump_img = JUMPING

        self.dino_duck = False
        self.dino_run = True
        self.dino_jump = False

        self.step_index = 0
        self.jump_vel = self.JUMP_VEL
        self.image = self.run_img[0]
        self.dino_rect = self.image.get_rect()
        self.dino_rect.x = self.X_POS
        self.dino_rect.y = self.Y_POS

    def update(self, userInput):
        if self.dino_duck:
            self.duck()
        if self.dino_run:
            self.run()
        if self.dino_jump:
            self.jump()

        if self.step_index >= 10:
            self.step_index = 0

        if userInput[pg.K_UP] and not self.dino_jump:
            self.dino_duck = False
            self.dino_run = False
            self.dino_jump = True
        elif userInput[pg.K_DOWN] and not self.dino_jump:
            self.dino_duck = True
            self.dino_run = False
            self.dino_jump = False
        elif not (self.dino_jump or userInput[pg.K_DOWN]):
            self.dino_duck = False
            self.dino_run = True
            self.dino_jump = False

    def duck(self):
        self.image = self.duck_img[self.step_index // 5]
        self.dino_rect = self.image.get_rect()
        self.dino_rect.x = self.X_POS
        self.dino_rect.y = self.Y_POS_DUCK
        self.step_index += 1

    def run(self):
        self.image = self.run_img[self.step_index // 5]
        self.dino_rect = self.image.get_rect()
        self.dino_rect.x = self.X_POS
        self.dino_rect.y = self.Y_POS
        self.step_index += 1

    def jump(self):
        self.image = self.jump_img
        if self.dino_jump:
            self.dino_rect.y -= self.jump_vel * 4
            self.jump_vel -= 0.8
        if self.jump_vel < -self.JUMP_VEL:
            self.dino_jump = False
            self.jump_vel = self.JUMP_VEL

    def draw(self, SCREEN):
        SCREEN.blit(self.image, (self.dino_rect.x, self.dino_rect.y))


def main():
    global game_speed, x_pos_bg, y_pos_bg, points, obstacles
    run = True
    clock = pg.time.Clock()
    player = Dinosaur()
    game_speed = 20
    x_pos_bg = 0
    y_pos_bg = 380
    points = 0
    font = pg.font.Font("freesansbold.ttf", 20)
    obstacles = []
    pg.display.set_caption("恐竜ゲーム")

    def score():
        global points, game_speed
        global points, game_speed
        points += 0.1
        if points % 100 == 0:
            game_speed += 1

        text = font.render(f"ScorePoint:{points:.0f}", True, (0, 0, 0))
        textRect = text.get_rect()
        textRect.center = (1000, 40)
        SCREEN.blit(text, textRect)

    def background():
        global x_pos_bg, y_pos_bg
        image_width = BG.get_width()
        SCREEN.blit(BG, (x_pos_bg, y_pos_bg))
        SCREEN.blit(BG, (image_width + x_pos_bg, y_pos_bg))
        if x_pos_bg <= -image_width:
            SCREEN.blit(BG, (image_width + x_pos_bg, y_pos_bg))
            x_pos_bg = 0
        x_pos_bg -= game_speed

    while run:
        for event in pg.event.get():
            if event.type == pg.QUIT:
                run = False
                sys.exit()

        SCREEN.fill((255, 255, 255))
        userInput = pg.key.get_pressed()

        player.draw(SCREEN)
        player.update(userInput)

        for obstacle in obstacles:
            obstacle.draw(SCREEN)
            obstacle.update()
            if player.dino_rect.colliderect(obstacle.rect):
                pg.time.delay(2000)

        background()
        score()
        clock.tick(30)
        pg.display.update()


if __name__ == "__main__":
    pg.init()
    main()
    pg.quit()
    sys.exit()
```