188 lines (152 loc) · 5.54 KB

Code Blame Raw

```python
import os
import random
import sys
import time
import pygame as pg


WIDTH = 1100
HEIGHT = 650
NUM_OF_BOMBS = 5
os.chdir(os.path.dirname(os.path.abspath(__file__)))


def check_bound(obj_rct: pg.Rect) -> tuple[bool, bool]:
    yoko, tate = True, True
    if obj_rct.left < 0 or WIDTH < obj_rct.right:
        yoko = False
    if obj_rct.top < 0 or HEIGHT < obj_rct.bottom:
        tate = False
    return yoko, tate


class Bird:
    delta = {
        pg.K_w: (0, -5),
        pg.K_s: (0, +5),
        pg.K_a: (-5, 0),
        pg.K_d: (+5, 0),
    }
    img0 = pg.transform.rotozoom(pg.image.load("fig/3.png"), 0, 0.9)
    img = pg.transform.flip(img0, True, False)
    imgs = {
        (+5, 0): img,
        (+5, -5): pg.transform.rotozoom(img, 45, 0.9),
        (0, -5): pg.transform.rotozoom(img, 90, 0.9),
        (-5, -5): pg.transform.rotozoom(img0, -45, 0.9),
```

```python
            (-5, 0): img0,
            (-5, +5): pg.transform.rotozoom(img0, 45, 0.9),
            (0, +5): pg.transform.rotozoom(img, -90, 0.9),
            (+5, +5): pg.transform.rotozoom(img, -45, 0.9),
        }

        def __init__(self, xy: tuple[int, int]):
            self.img = __class__.imgs[(+5, 0)]
            self.rct = self.img.get_rect()
            self.rct.center = xy

        def change_img(self, num: int, screen: pg.Surface):
            self.img = pg.transform.rotozoom(pg.image.load(f"fig/{num}.png"), 0, 0.9)
            screen.blit(self.img, self.rct)

        def update(self, key_lst: list[bool], screen: pg.Surface):
            sum_mv = [0, 0]
            for k, mv in __class__.delta.items():
                if key_lst[k]:
                    sum_mv[0] += mv[0]
                    sum_mv[1] += mv[1]
            self.rct.move_ip(sum_mv)
            if check_bound(self.rct) != (True, True):
                self.rct.move_ip(-sum_mv[0], -sum_mv[1])
            if not (sum_mv[0] == 0 and sum_mv[1] == 0):
                self.img = __class__.imgs[tuple(sum_mv)]
            screen.blit(self.img, self.rct)


    class Beam:
        def __init__(self, bird: "Bird"):
            self.img = pg.image.load("fig/beam.png")
            self.rct = self.img.get_rect()
            self.rct.centery = bird.rct.centery
            self.rct.left = bird.rct.right
            self.vx, self.vy = +5, 0   # 速めてもOK

        def update(self, screen: pg.Surface):
            self.rct.move_ip(self.vx, self.vy)
            screen.blit(self.img, self.rct)

        def is_out(self) -> bool:
            """画面外判定"""
            yoko, tate = check_bound(self.rct)
            return not (yoko and tate)


    class Bomb:
        def __init__(self, color: tuple[int, int, int], rad: int):
            self.img = pg.Surface((2*rad, 2*rad))
            pg.draw.circle(self.img, color, (rad, rad), rad)
            self.img.set_colorkey((0, 0, 0))
```

```python
            self.rct = self.img.get_rect()
            self.rct.center = random.randint(0, WIDTH), random.randint(0, HEIGHT)
            self.vx, self.vy = +5, +5

        def update(self, screen: pg.Surface):
            yoko, tate = check_bound(self.rct)
            if not yoko:
                self.vx *= -1
            if not tate:
                self.vy *= -1
            self.rct.move_ip(self.vx, self.vy)
            screen.blit(self.img, self.rct)


    class Score:
        def __init__(self):
            self.fonto = pg.font.SysFont("hgp創英角ﾎﾟｯﾌﾟ体", 30)
            self.color = (0, 0, 255)
            self.scr = 0
            self.img = self.fonto.render("表示させる文字 ", 0, self.color)
            self.rct = self.img.get_rect()
            self.rct.center = 100, HEIGHT-50

        def update(self, screen: pg.Surface):
            self.img = self.fonto.render(f"スコア:{self.scr}", 0, self.color)
            screen.blit(self.img, self.rct)


    def main():
        pg.display.set_caption("たたかえ！こうかとん")
        screen = pg.display.set_mode((WIDTH, HEIGHT))
        bg_img = pg.image.load("fig/pg_bg.jpg")

        bird = Bird((300, 200))
        bombs = [Bomb((255, 0, 0), 10) for _ in range(NUM_OF_BOMBS)]
        beams = []  # ← Beam リスト

        clock = pg.time.Clock()
        score = Score()
        tmr = 0

        while True:
            for event in pg.event.get():
                if event.type == pg.QUIT:
                    return
                if event.type == pg.KEYDOWN and event.key == pg.K_RSHIFT:
                    beams.append(Beam(bird))  # ← 追加

            screen.blit(bg_img, [0, 0])

            # こうかとん死亡判定
            for bomb in bombs:
```

```python
                if bird.rct.colliderect(bomb.rct):
                    bird.change_img(8, screen)
                    fonto = pg.font.Font(None, 80)
                    txt = fonto.render("Game Over", True, (255, 0, 0))
                    screen.blit(txt, [WIDTH//2-150, HEIGHT//2])

                    pg.display.update()
                    time.sleep(1)
                    return

            # ビームと爆弾の衝突判定
            for i, beam in enumerate(beams):
                for j, bomb in enumerate(bombs):
                    if beam is not None and bomb is not None:
                        if beam.rct.colliderect(bomb.rct):
                            beams[i] = None
                            bombs[j] = None
                            score.scr += 1

            # None を削除
            beams = [b for b in beams if b is not None]
            bombs = [b for b in bombs if b is not None]

            # ビームが画面外に出たらリストから削除
            beams = [b for b in beams if not b.is_out()]

            key_lst = pg.key.get_pressed()
            bird.update(key_lst, screen)

            # ビーム描画
            for beam in beams:
                beam.update(screen)

            # 爆弾描画
            for bomb in bombs:
                bomb.update(screen)

            score.update(screen)
            pg.display.update()
            tmr += 1
            clock.tick(50)


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