ProjExD_3 / fight_kokaton.py

c0c24020ae multibeamブランチ 8055167 · 37 minutes ago

197 lines (174 loc) · 7 KB

Code Blame Raw

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


WIDTH = 1100  # ゲームウィンドウの幅
HEIGHT = 650  # ゲームウィンドウの高さ
NUM_OF_BOMBS = 5  # 爆弾の個数
os.chdir(os.path.dirname(os.path.abspath(__file__)))


def check_bound(obj_rct: pg.Rect) -> tuple[bool, bool]:
    """
    オブジェクトが画面内or画面外を判定し，真理値タプルを返す関数
    引数：こうかとんや爆弾，ビームなどのRect
    戻り値：横方向，縦方向のはみ出し判定結果（画面内：True／画面外：False）
    """
    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:
    """
    ゲームキャラクター（こうかとん）に関するクラス
    """
    delta = {  # 押下キーと移動量の辞書
        pg.K_UP: (0, -5),
        pg.K_DOWN: (0, +5),
        pg.K_LEFT: (-5, 0),
        pg.K_RIGHT: (+5, 0),
    }
    img0 = pg.transform.rotozoom(pg.image.load("fig/3.png"), 0, 0.9)
    img = pg.transform.flip(img0, True, False)  # デフォルトのこうかとん（右向き）
    imgs = {  # 0度から反時計回りに定義
        (+5, 0): img,  # 右
        (+5, -5): pg.transform.rotozoom(img, 45, 0.9),  # 右上
```

```python
            (0, -5): pg.transform.rotozoom(img, 90, 0.9),  # 上
            (-5, -5): pg.transform.rotozoom(img0, -45, 0.9),  # 左上
            (-5, 0): img0,  # 左
            (-5, +5): pg.transform.rotozoom(img0, 45, 0.9),  # 左下
            (0, +5): pg.transform.rotozoom(img, -90, 0.9),  # 下
            (+5, +5): pg.transform.rotozoom(img, -45, 0.9),  # 右下
        }

    def __init__(self, xy: tuple[int, int]):
        """
        こうかとん画像Surfaceを生成する
        引数 xy：こうかとん画像の初期位置座標タプル
        """
        self.img = __class__.imgs[(+5, 0)]
        self.rct: pg.Rect = self.img.get_rect()
        self.rct.center = xy

    def change_img(self, num: int, screen: pg.Surface):
        """
        こうかとん画像を切り替え，画面に転送する
        引数1 num：こうかとん画像ファイル名の番号
        引数2 screen：画面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):
        """
        押下キーに応じてこうかとんを移動させる
        引数1 key_lst：押下キーの真理値リスト
        引数2 screen：画面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:
    """
    こうかとんが放つビームに関するクラス
    """
    def __init__(self, bird:"Bird"):
        """
        ビーム画像Surfaceを生成する
        引数 bird：ビームを放つこうかとん（Birdインスタンス）
        """
        self.img = pg.image.load(f"fig/beam.png")  # こうかとんSurface
        self.rct = self.img.get_rect()  # こうかとんRect
        self.rct.centery = bird.rct.centery  # こうかとんの中心縦座標
        self.rct.left = bird.rct.right  # ビームの左座標 = こうかとんの右座標
```

```python
            self.vx, self.vy = +5, 0

        def update(self, screen: pg.Surface):
            """
            ビームを速度ベクトルself.vx, self.vyに基づき移動させる
            引数 screen：画面Surface
            """
            if check_bound(self.rct) == (True, True):
                self.rct.move_ip(self.vx, self.vy)
                screen.blit(self.img, self.rct)


    class Bomb:
        """
        爆弾に関するクラス
        """
        def __init__(self, color: tuple[int, int, int], rad: int):
            """
            引数に基づき爆弾円Surfaceを生成する
            引数1 color：爆弾円の色タプル
            引数2 rad：爆弾円の半径
            """
            self.img = pg.Surface((2*rad, 2*rad))
            pg.draw.circle(self.img, color, (rad, rad), rad)
            self.img.set_colorkey((0, 0, 0))
            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):
            """
            爆弾を速度ベクトルself.vx, self.vyに基づき移動させる
            引数 screen：画面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)


    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))
        beam = None
        # bomb = Bomb((255, 0, 0), 10)
        bombs = [Bomb((255, 0, 0), 10) for _ in range(NUM_OF_BOMBS)]
        clock = pg.time.Clock()
        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_SPACE:
```

```python
                    # スペースキー押下でBeamクラスのインスタンス生成
                    beam = Beam(bird)
            screen.blit(bg_img, [0, 0])


            # if bomb is not None:
            for bomb in bombs:
                if bird.rct.colliderect(bomb.rct):
                    # ゲームオーバー時に，こうかとん画像を切り替え，1秒間表示させる
                    bird.change_img(8, screen)
                    pg.display.update()
                    time.sleep(1)
                    return

           #if beam is not None:
            for j, bomb in enumerate(bombs):
                if beam is not None:
                    if beam.rct.colliderect(bomb.rct):  # ビームと爆弾の衝突判定
                        beam = None  # ビームを消す
                        bombs[j] = None  # 爆弾を消す
                        bird.change_img(6, screen)  # よろこびエフェクト
                bombs = [bomb for bomb in bombs if bomb is not None]  # 撃ち落とされてない爆弾だけのリストに

            key_lst = pg.key.get_pressed()
            bird.update(key_lst, screen)
            if beam is not None:
                beam.update(screen)
            for bomb in bombs:
                bomb.update(screen)
            pg.display.update()
            tmr += 1
            clock.tick(50)


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

c0c24020ae / ProjExD_3

Code | Issues 1 | Pull requests | Actions | Projects | Wiki | Security | Insight

Branches

New branch

Overview | Yours | Active | Stale | All

Search branches...

Default

Branch	Updated	Check status	Behind	Ahead	Pull request
main	3 hours ago		Default		

Your branches

Branch	Updated	Check status	Behind	Ahead	Pull request
issue2	7 minutes ago		0	1	
multibeam	38 minutes ago		0	1	
score	42 minutes ago		1	2	
beam	2 hours ago		1	0	
happy	2 hours ago		1	1	

Active branches

Branch	Updated	Check status	Behind	Ahead	Pull request
issue2	7 minutes ago		0	1	
multibeam	38 minutes ago		0	1	
score	42 minutes ago		1	2	
beam	2 hours ago		1	0	
happy	2 hours ago		1	1	

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks or learn more about diff comparisons.

base: main ← compare: multibeam ✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. Learn about pull requests

Create pull request

1 commit 1 file changed 1 contributor

Commits on Apr 29, 2025

multibeamブランチ

c0c24020ae committed 39 minutes ago

1 8055167

Showing 1 changed file with 1 addition and 0 deletions.

Split Unified

1 fight_kokaton.py

```
160 160                     beam = Beam(bird)
161 161             screen.blit(bg_img, [0, 0])
162 162
    163 +
163 164             # if bomb is not None:
164 165             for bomb in bombs:
165 166                 if bird.rct.colliderect(bomb.rct):
```