

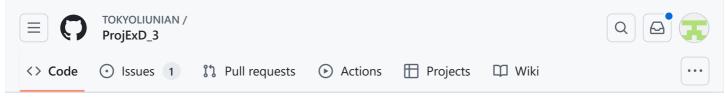
189 lines (173 loc) \cdot 6.45 KB

```
Raw 📮 🕹
                                                                        83
                                                                                                    <>
Code
        Blame
    1
          import os
    2
          import random
    3
          import sys
    4
          import time
    5
          import pygame as pg
    6
    7
    8
          WIDTH = 1100 # ゲームウィンドウの幅
          HEIGHT = 650
    9
   10
          os.chdir(os.path.dirname(os.path.abspath(__file__)))
   11
   12
         def check_bound(obj_rct: pg.Rect) -> tuple[bool, bool]:
   13
   14
             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
   16
             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)
   17
   18
   19
             yoko, tate = True, True
   20
             if obj_rct.left < 0 or WIDTH < obj_rct.right:</pre>
   21
                 yoko = False
             if obj_rct.top < 0 or HEIGHT < obj_rct.bottom:</pre>
   22
                 tate = False
   23
   24
             return yoko, tate
   25
   26
   27
         class Bird:
   28
             ゲームキャラクター(こうかとん)に関するクラス
   29
   30
   31
             delta = { # 押下キーと移動量の辞書
                 pg.K_UP: (0, -5),
   32
                 pg.K_DOWN: (0, +5),
   33
                 pg.K_LEFT: (-5, 0),
   34
   35
                 pg.K_RIGHT: (+5, 0),
   36
             img0 = pg.transform.rotozoom(pg.image.load("fig/3.png"), 0, 0.9)
   37
             img = pg.transform.flip(img0, True, False) # デフォルトのこうかとん (右向き)
   38
             imgs = { # 0度から反時計回りに定義
   39
   40
                 (+5, 0): img, # 右
   41
                 (+5, -5): pg.transform.rotozoom(img, 45, 0.9), # 右上
                 (0, -5): pg.transform.rotozoom(img, 90, 0.9), # 上
```

```
(-5, -5): pg.transform.rotozoom(img0, -45, 0.9), # 左上
43
44
              (-5, 0): img0, #左
45
              (-5, +5): pg.transform.rotozoom(img0, 45, 0.9), # 左下
              (0, +5): pg.transform.rotozoom(img, -90, 0.9), # 下
46
47
              (+5, +5): pg.transform.rotozoom(img, -45, 0.9), # 右下
48
          }
49
50
           def __init__(self, xy: tuple[int, int]):
51
52
               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
53
54
55
              self.img = __class__.imgs[(+5, 0)]
56
              self.rct: pg.Rect = self.img.get_rect()
57
              self.rct.center = xy
58
          def change_img(self, num: int, screen: pg.Surface):
59
60
               こうかとん画像を切り替え,画面に転送する
61
62
              引数1 num: こうかとん画像ファイル名の番号
63
              引数2 screen: 画面Surface
64
              self.img = pg.transform.rotozoom(pg.image.load(f"fig/{num}.png"), 0, 0.9)
65
              screen.blit(self.img, self.rct)
66
67
68
           def update(self, key_lst: list[bool], screen: pg.Surface):
69
              押下キーに応じてこうかとんを移動させる
70
              引数1 key lst: 押下キーの真理値リスト
71
72
              引数2 screen: 画面Surface
73
74
              sum_mv = [0, 0]
75
              for k, mv in __class__.delta.items():
                  if key_lst[k]:
76
77
                      sum mv[0] += mv[0]
                      sum_mv[1] += mv[1]
78
79
              self.rct.move ip(sum mv)
20
              if check_bound(self.rct) != (True, True):
                  self.rct.move_ip(-sum_mv[0], -sum_mv[1])
81
82
              if not (sum_mv[0] == 0 and sum_mv[1] == 0):
83
                  self.img = class .imgs[tuple(sum mv)]
              screen.blit(self.img, self.rct)
84
       #从这以下是Beam
85
       class Beam:
86
87
           こうかとんが放つビームに関するクラス
88
89
90
           def __init__(self, bird: "Bird"):
91
92
              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)
93
94
95
              self.img = pg.image.load("fig/beam.png")
              self.rct = self.img.get_rect()
96
97
              self.rct.centery = bird.rct.centery
98
              self.rct.left = bird.rct.right
99
              self.vx, self.vy = +5, 0
```

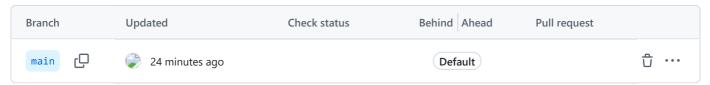
```
101 🗸
            def update(self, screen: pg.Surface):
102
                .....
                ビームを速度ベクトルself.vx, self.vyに基づき移動させる
103
                引数 screen: 画面Surface
104
105
                if check_bound(self.rct) == (True, True):
106
107
                    self.rct.move_ip(self.vx, self.vy)
108
                    screen.blit(self.img, self.rct)
        #以上是Beam
109
110
        class Bomb:
111
            爆弾に関するクラス
112
            ....
113
114 🗸
            def __init__(self, color: tuple[int, int, int], rad: int):
115
                引数に基づき爆弾円Surfaceを生成する
116
                引数1 color: 爆弾円の色タプル
117
                引数2 rad: 爆弾円の半径
118
                0.00
119
120
                self.img = pg.Surface((2*rad, 2*rad))
121
                pg.draw.circle(self.img, color, (rad, rad), rad)
                self.img.set_colorkey((0, 0, 0))
122
                self.rct = self.img.get_rect()
123
124
                self.rct.center = random.randint(0, WIDTH), random.randint(0, HEIGHT)
125
                self.vx, self.vy = +5, +5
126
127 🗸
            def update(self, screen: pg.Surface):
128
129
                爆弾を速度ベクトルself.vx, self.vyに基づき移動させる
130
                引数 screen: 画面Surface
131
                yoko, tate = check_bound(self.rct)
132
               if not yoko:
133
                   self.vx *= -1
134
135
                if not tate:
                    self.vy *= -1
136
                self.rct.move_ip(self.vx, self.vy)
137
138
                screen.blit(self.img, self.rct)
139
140
141 ∨ def main():
142
            pg.display.set_caption("たたかえ! こうかとん")
143
            screen = pg.display.set_mode((WIDTH, HEIGHT))
            bg_img = pg.image.load("fig/pg_bg.jpg")
144
145
            bird = Bird((300, 200))
            bomb = Bomb((255, 0, 0), 10)
146
147
            clock = pg.time.Clock()
148
            tmr = 0
            while True:
149
                for event in pg.event.get():
150
151
                    if event.type == pg.QUIT:
152
153
                    if event.type == pg.KEYDOWN and event.key == pg.K_SPACE:
                        #按下空格键Beam生成
154
155
                        beam = Beam(bird)
156
                screen.blit(bg_img, [0, 0])
157
158
                if bomb is not None:
```

```
159
                   if bird.rct.colliderect(bomb.rct):
160
                   #游戏结束时,切换图像,显示1秒
161
                       bird.change_img(8, screen)
162
                       fonto = pg.font.Font(None, 80)
                       txt = fonto.render("Game Over", True, (255, 0, 0))
163
                       screen.blit(txt, [WIDTH//2-150, HEIGHT//2])
164
165
                       pg.display.update()
                       time.sleep(1)
166
167
                       return
168
               if beam is not None:
169
170
                   if bomb is not None:
                       if beam.rct.colliderect(bomb.rct):#后加的
171
172
                           beam = None
                           bomb = None # 爆弾消失
173
                           bird.change_img(6, screen)#よろこびエフェクト
174
175
               key_lst = pg.key.get_pressed()
               bird.update(key_lst, screen)
176
177
               if beam is not None:
178
                   beam.update(screen)
               bomb.update(screen)
179
               pg.display.update()
180
181
               tmr += 1
182
               clock.tick(50)
183
        # こうかとんの移動速度を調整するための時間調整
184
        if __name__ == "__main__":
185
186
           pg.init()
187
           main()
188
            pg.quit()
189
            sys.exit()
```

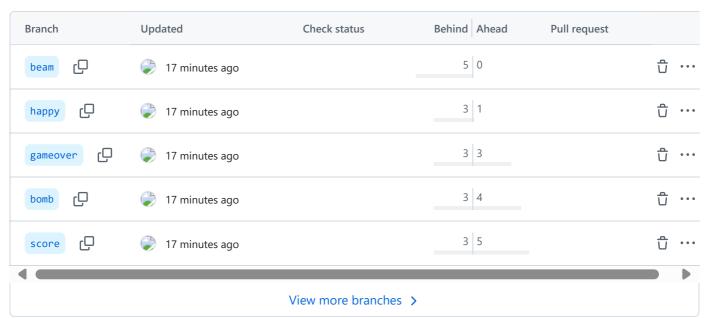


Branches Overview Yours Active Stale All Q Search branches...

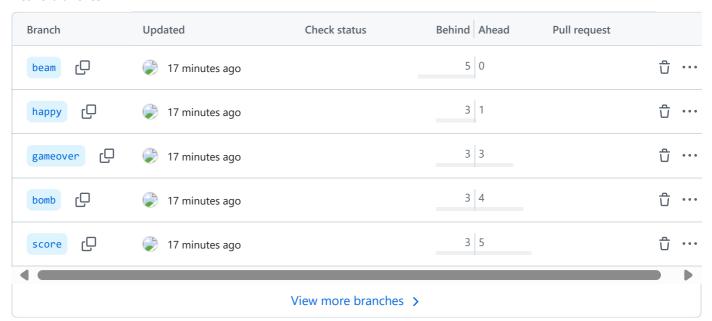
Default

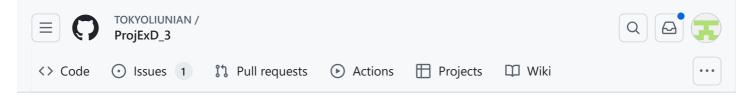


Your branches



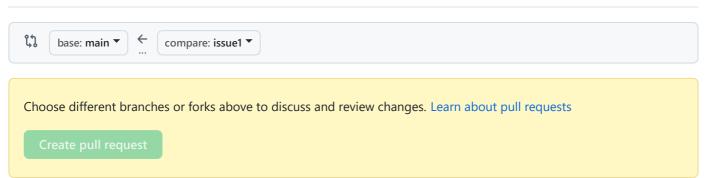
Active branches





Compare changes

Compare changes across branches, commits, tags, and more below. If you need to, you can also compare across forks.



There isn't anything to compare.

We couldn't figure out how to compare these references, do they point to valid commits?

