


 TOKYOLIUNIAN /
ProjExD_3



[Code](#) [Issues](#) [1](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [...](#)

  bomb ▾ ProjExD_3 / fight_kokaton.py 

Go to file

t

...

 TOKYO_LIUNIAN 练习5

c3314c9 · 2 hours ago



189 lines (173 loc) · 6.45 KB

[Code](#) [Blame](#)

 [Raw](#)     

```
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
9  HEIGHT = 650
10 os.chdir(os.path.dirname(os.path.abspath(__file__)))
11
12
13  def check_bound(obj_rct: pg.Rect) -> tuple[bool, bool]:
14      """
15      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
16      引数: こうかとんや爆弾、ビームなどのRect
17      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）
18      """
19      yoko, tate = True, True
20      if obj_rct.left < 0 or WIDTH < obj_rct.right:
21          yoko = False
22      if obj_rct.top < 0 or HEIGHT < obj_rct.bottom:
23          tate = False
24      return yoko, tate
25
26
27  class Bird:
28      """
29      ゲームキャラクター（こうかとん）に関するクラス
30      """
31      delta = { # 押下キーと移動量の辞書
32          pg.K_UP: (0, -5),
33          pg.K_DOWN: (0, +5),
34          pg.K_LEFT: (-5, 0),
35          pg.K_RIGHT: (+5, 0),
36      }
37      img0 = pg.transform.rotozoom(pg.image.load("fig/3.png"), 0, 0.9)
38      img = pg.transform.flip(img0, True, False) # デフォルトのこうかとん（右向き）
39      imgs = { # 0度から反時計回りに定義
40          (+5, 0): img, # 右
41          (+5, -5): pg.transform.rotozoom(img, 45, 0.9), # 右上
42          (0, -5): pg.transform.rotozoom(img, 90, 0.9), # 上
```

```

43         (-5, -5): pg.transform.rotozoom(img0, -45, 0.9), # 左上
44         (-5, 0): img0, # 左
45         (-5, +5): pg.transform.rotozoom(img0, 45, 0.9), # 左下
46         (0, +5): pg.transform.rotozoom(img, -90, 0.9), # 下
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
53         引数 xy: こうかとん画像の初期位置座標タプル
54         """
55         self.img = __class__.imgs[(+5, 0)]
56         self.rct: pg.Rect = self.img.get_rect()
57         self.rct.center = xy
58
59     ✓ def change_img(self, num: int, screen: pg.Surface):
60         """
61         こうかとん画像を切り替え、画面に転送する
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         """
65         self.img = pg.transform.rotozoom(pg.image.load(f"fig/{num}.png"), 0, 0.9)
66         screen.blit(self.img, self.rct)
67
68     ✓ def update(self, key_lst: list[bool], screen: pg.Surface):
69         """
70         押下キーに応じてこうかとんを移動させる
71         引数1 key_lst: 押下キーの真理値リスト
72         引数2 screen: 画面Surface
73         """
74         sum_mv = [0, 0]
75         for k, mv in __class__.delta.items():
76             if key_lst[k]:
77                 sum_mv[0] += mv[0]
78                 sum_mv[1] += mv[1]
79         self.rct.move_ip(sum_mv)
80         if check_bound(self.rct) != (True, True):
81             self.rct.move_ip(-sum_mv[0], -sum_mv[1])
82         if not (sum_mv[0] == 0 and sum_mv[1] == 0):
83             self.img = __class__.imgs[tuple(sum_mv)]
84         screen.blit(self.img, self.rct)
85         # 从这以下是Beam
86     ✓ class Beam:
87         """
88         こうかとんが放つビームに関するクラス
89         """
90     ✓ def __init__(self, bird: "Bird"):
91         """
92         ビーム画像Surfaceを生成する
93         引数 bird: ビームを放つこうかとん (Birdインスタンス)
94         """
95         self.img = pg.image.load("fig/beam.png")
96         self.rct = self.img.get_rect()
97         self.rct.centery = bird.rct.centery
98         self.rct.left = bird.rct.right
99         self.vx, self.vy = +5, 0
100

```

```
101  ✓    def update(self, screen: pg.Surface):
102         """
103         ビームを速度ベクトルself.vx, self.vyに基づき移動させる
104         引数 screen: 画面Surface
105         """
106         if check_bound(self.rct) == (True, True):
107             self.rct.move_ip(self.vx, self.vy)
108             screen.blit(self.img, self.rct)
109     #以上はBeam
110  ✓    class Bomb:
111         """
112         爆弾に関するクラス
113         """
114  ✓    def __init__(self, color: tuple[int, int, int], rad: int):
115         """
116         引数に基づき爆弾円Surfaceを生成する
117         引数1 color: 爆弾円の色タプル
118         引数2 rad: 爆弾円の半径
119         """
120         self.img = pg.Surface((2*rad, 2*rad))
121         pg.draw.circle(self.img, color, (rad, rad), rad)
122         self.img.set_colorkey((0, 0, 0))
123         self.rct = self.img.get_rect()
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         """
132         yoko, tate = check_bound(self.rct)
133         if not yoko:
134             self.vx *= -1
135         if not tate:
136             self.vy *= -1
137         self.rct.move_ip(self.vx, self.vy)
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))
144         bg_img = pg.image.load("fig/pg_bg.jpg")
145         bird = Bird((300, 200))
146         bomb = Bomb((255, 0, 0), 10)
147         clock = pg.time.Clock()
148         tmr = 0
149         while True:
150             for event in pg.event.get():
151                 if event.type == pg.QUIT:
152                     return
153                 if event.type == pg.KEYDOWN and event.key == pg.K_SPACE:
154                     #按下空格键Beam生成
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)
163             txt = fonto.render("Game Over", True, (255, 0, 0))
164             screen.blit(txt, [WIDTH//2-150, HEIGHT//2])
165             pg.display.update()
166             time.sleep(1)
167             return
168
169         if beam is not None:
170             if bomb is not None:
171                 if beam.rct.colliderect(bomb.rct):#后加的
172                     beam = None
173                     bomb = None # 爆彈消失
174                     bird.change_img(6, screen)#よろこびエフェクト
175             key_lst = pg.key.get_pressed()
176             bird.update(key_lst, screen)
177             if beam is not None:
178                 beam.update(screen)
179             bomb.update(screen)
180             pg.display.update()
181             tmr += 1
182             clock.tick(50)
183
184     # こうかとの移動速度を調整するための時間調整
185     if __name__ == "__main__":
186         pg.init()
187         main()
188         pg.quit()
189         sys.exit()
```