


☰

 Takuma22154 / ProjExD_03

🔍

+

🕒

🔗

📁

🌐

<> Code

🕒 Issues 1

🔗 Pull requests

🎬 Actions

📁 Projects

📖 Wiki

🛡️ Security

📈 Insights

⚙️ Settings


📁 master

ProjExD_03 / fight_kokaton.py

🔍 Go to file

t

⋮

 Takuma22154 演習問題1 : 一瞬だけ出現,追加でコメント調整

31 minutes ago

🗨️

🕒

223 lines (193 loc) · 7.38 KB

Code

Blame

Raw

📄

📥

✎

⌵

🔗

```
1 import random
2 import sys
3 import time
4
5 import pygame as pg
6
7
8 WIDTH = 1600 # ゲームウィンドウの幅
9 HEIGHT = 900 # ゲームウィンドウの高さ
10 NUM_OF_BOMBS = 5
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, または、爆弾SurfaceのRect
17     戻り値: 横方向, 縦方向のみ出し判定結果 (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
38     def __init__(self, num: int, xy: tuple[int, int]):
39         """
40         こうかとん画像Surfaceを生成する
41         引数1 num: こうかとん画像ファイル名の番号
42         引数2 xy: こうかとん画像の位置座標タプル
43         """
44         img0 = pg.transform.rotozoom(pg.image.load(f"ex03/fig/{num}.png"), 0, 2.0)
45         img = pg.transform.flip(img0, True, False)
46         self.imgs = {
47             (+5, 0): img,
48             (+5, -5): pg.transform.rotozoom(img, 45, 1.0),
49             (0, -5): pg.transform.rotozoom(img, 90, 1.0),
50             (-5, -5): pg.transform.rotozoom(img0, -45, 1.0),
51             (-5, 0): img0,
52             (-5, +5): pg.transform.rotozoom(img0, 45, 1.0),
53             (0, +5): pg.transform.rotozoom(img, -90, 1.0),
54             (+5, +5): pg.transform.rotozoom(img, -45, 1.0),
55         }
56         self.img = self.imgs[+5, 0]
57         self.rct = self.img.get_rect()
58         self.rct.center = xy
59
60     def change_img(self, num: int, screen: pg.Surface):
```

https://github.com/Takuma22154/ProjExD_03/blob/master/fight_kokaton.py

1/4

```

60     def change_img(self, num: int, screen: pg.Surface):
61         """
62         こうかとん画像を切り替え、画面に転送する
63         引数1 num: こうかとん画像ファイル名の番号
64         引数2 screen: 画面Surface
65         """
66         self.img = pg.transform.rotozoom(pg.image.load(f"ex03/fig/{num}.png"), 0, 2.0)
67         screen.blit(self.img, self.rct)
68
69     def update(self, key_lst: list[bool], screen: pg.Surface):
70         """
71         押下キーに応じてこうかとんを移動させる
72         引数1 key_lst: 押下キーの真値リスト
73         引数2 screen: 画面Surface
74         """
75         sum_mv = [0, 0]
76         for k, mv in __class__.delta.items():
77             if key_lst[k]:
78                 sum_mv[0] += mv[0]
79                 sum_mv[1] += mv[1]
80         self.rct.move_ip(sum_mv)
81         if check_bound(self.rct) != (True, True):
82             self.rct.move_ip(-sum_mv[0], -sum_mv[1])
83         if not (sum_mv[0] == 0 and sum_mv[1] == 0):
84             self.img = self.imgs[tuple(sum_mv)]
85         screen.blit(self.img, self.rct)
86
87
88     class Bomb:
89         """
90         爆弾に関するクラス
91         """
92     def __init__(self, color: tuple[int, int, int], rad: int):
93         """
94         引数に基づき爆弾円Surfaceを生成する
95         引数1 color: 爆弾円の色タプル
96         引数2 rad: 爆弾円の半径
97         """
98         self.img = pg.Surface((2*rad, 2*rad))
99         pg.draw.circle(self.img, color, (rad, rad), rad)
100         self.img.set_colorkey((0, 0, 0))
101         self.rct = self.img.get_rect()
102         self.rct.center = random.randint(0, WIDTH), random.randint(0, HEIGHT)
103         self.vx, self.vy = +5, +5
104
105     def update(self, screen: pg.Surface):
106         """
107         爆弾を速度ベクトルself.vx, self.vyに基づき移動させる
108         引数 screen: 画面Surface
109         """
110         yoko, tate = check_bound(self.rct)
111         if not yoko:
112             self.vx *= -1
113         if not tate:
114             self.vy *= -1
115         self.rct.move_ip(self.vx, self.vy)
116         screen.blit(self.img, self.rct)
117
118
119     class Beam:
120         """
121         こうかとんが射出するビームに関するクラス
122         """
123     def __init__(self, bird: Bird): #Bird必須かも
124         """
125         引数に基づきビームSurfaceを生成する
126         引数 bird: ビームを放つこうかとん
127         """
128         self.img = pg.image.load(f"ex03/fig/beam.png")
129         self.rct = self.img.get_rect()
130         self.rct.left = bird.rct.right
131         self.rct.centery = bird.rct.centery
132         self.vx, self.vy = +5, 0
133
134     def update(self, screen: pg.Surface):
135         """

```

```

136     ビームを速度ベクトルself.vx, self.vyに基づき移動させる
137     引数 screen : 画面Surface
138     """
139     self.rct.move_ip(self.vx, self.vy)
140     screen.blit(self.img, self.rct)
141
142
143  class Explosion:
144     """
145     爆弾を撃ち落とした時のエフェクト表示に関するクラス
146     """
147  def __init__(self, bomb:Bomb):
148     exp_img = pg.transform.flip(pg.image.load(f"ex03/fig/explosion.gif"), False, False) # 爆発の画像の挿入
149     self.imgs = [exp_img, pg.transform.flip(exp_img, True, True)]
150     self.rct = exp_img.get_rect()
151     self.rct.x = bomb.rct.centerx
152     self.rct.y = bomb.rct.centery
153     self.life = 100
154
155  def update(self, screen, life): # 時間に応じた画像の変更（未動作）
156     if life >= 50:
157         screen.blit(self.imgs[0], self.rct)
158     else:
159         screen.blit(self.imgs[1], self.rct)
160
161
162
163  def main():
164     pg.display.set_caption("たたかえ！こうかとん")
165     screen = pg.display.set_mode((WIDTH, HEIGHT))
166     bg_img = pg.image.load("ex03/fig/pg_bg.jpg")
167     bird = Bird(3, (900, 400))
168     #bomb = Bomb((255, 0, 0), 10)
169     bombs = [Bomb((255, 0, 0), 10) for _ in range(NUM_OF_BOMBS)]
170     exps = []
171     beam = None
172     life = 100
173
174     clock = pg.time.Clock()
175     tmr = 0
176     while True:
177         for event in pg.event.get():
178             if event.type == pg.QUIT:
179                 return
180             if event.type == pg.KEYDOWN and event.key == pg.K_SPACE:
181                 beam = Beam(bird) #Beamクラスのインスタンスを呼ぶ
182                 screen.blit(bg_img, [0, 0])
183
184         for bomb in bombs:
185             if bird.rct.colliderect(bomb.rct):
186                 # 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
187                 bird.change_img(8, screen)
188                 pg.display.update()
189                 time.sleep(1)
190                 return
191
192         for i, bomb in enumerate(bombs):
193             if beam != None:
194                 if bomb.rct.colliderect(beam.rct): # 爆弾を撃ち落とした時の処理
195                     exps.append(Explosion(bombs[i])) # エフェクトのリストへの格納
196                     while life >= 0: # 199行目まで画像の変更の呼び出し（未動作）
197                         life -= 1
198                         exps[-1].update(screen, life)
199                     life += 100
200                     bombs[i] = None
201                     beam = None
202                     bird.change_img(6, screen)
203                     pg.display.update()
204
205         key_lst = pg.key.get_pressed()
206         bird.update(key_lst, screen)
207         bombs = [bomb for bomb in bombs if bomb != None]
208         for bomb in bombs:
209             bomb.update(screen)
210         if beam != None:
211             beam.update(screen)

```

```
212         pg.display.update()
213         tmr += 1
214         clock.tick(50)
215
216
217
218
219 if __name__ == "__main__":
220     pg.init()
221     main()
222     pg.quit()
223     sys.exit()
```