




 c0a2403829 / ProjExD_Group04




[Code](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

 C0A24038/speed ▾

ProjExD_Group04 / README.md



 Go to file t ...

 c0a2403829 name add

6851ed0 · last week 

34 lines (27 loc) · 1.28 KB

Preview Code Blame

Raw    ▾ 

丸焼き豪華豚

実行環境の必要条件

- python >= 3.10
- pygame >= 2.1

ゲームの概要

- こうかとんを振ってくる障害物から避け一定時間生き延びる
- 参考URL: [サイトタイトル](#)

ゲームの遊び方

- 矢印キーでこうかとんを操作する
- 2分以上生きれたらゲームクリア

ゲームの実装

共通基本機能

- 背景画像と主人公キャラクターの描画
- こうかとんの移動

分担追加機能

- 時間経過でサイズと速度が増加する、障害物の速度が増加する 宇都宮恵大
- 衝突判定、障害物に当たるとゲームオーバー
- 障害物の配置 (3, 4種類くらい)
- 時間の計測と表示、スタートとクリア画面 藤



- 有利、不利になるアイテムの配置、効果設定




ToDo

- ☐ ほげほげ機能
- ☐ ふがふが関数内の変数名の統一

メモ

- スタートとクリア画面、アイテム取得時に効果音を出す
- クラス内の変数は、すべて、「get_変数名」という名前のメソッドを介してアクセスするように設計してある
- すべてのクラスに関係する関数は、クラスの外で定義してある

 c0a2403829 / ProjExD_Group04



[Code](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

 C0A24038/speed ▾

ProjExD_Group04 / game.py



 c0a2403829 とうかとんが10秒ごとに早くでかくなる2

030431c · last week



239 lines (202 loc) · 7.69 KB

Code Blame

Raw Copy Download Edit More

```
1  import os
2  import random
3  import sys
4  import time
5  import pygame as pg
6
7
8
9  WIDTH, HEIGHT = 1100, 650
10  ▾ DELTA = { # 辞書の作成
11      # pg.K_UP: (0,-5),
12      # pg.K_DOWN: (0,+5),
13      pg.K_LEFT: (-5,0),
14      pg.K_RIGHT: (+5,0),
15  }
16  os.chdir(os.path.dirname(os.path.abspath(__file__)))
17
18  ▾ def check_bound(rct: pg.Rect) -> tuple[bool, bool]:
19      """
20      引数：こうかとんRectまたは爆弾Rect
21      戻り値：判定結果タプル（横、縦）
22      画面内ならTrue、画面外ならFalse
23      """
24      yoko, tate = True, True # 横、縦方向用の変数
25      # 横方向判定
26      if rct.left < 0 or WIDTH < rct.right: # 画面内だったら
27          yoko = False
28      if rct.top < 0 or HEIGHT < rct.bottom: # 画面外だったら
29          tate = False
30      return yoko, tate
31
32  ▾ def gameover(screen: pg.Surface) -> None: # ゲームオーバー機能
33      bg_rct = pg.Surface((WIDTH, HEIGHT))
34      pg.draw.rect(bg_rct,(0,0,0),(0,0,WIDTH,HEIGHT))
35      bg_rct.set_alpha(150)
36      screen.blit(bg_rct,[0, 0])
37      fonto = pg.font.Font(None, 70)
38      txt = fonto.render("Game Over",True, (255, 255, 255))
39      screen.blit(txt, [450, 325])
40      ck_img =pg.image.load("fig/4.png")
41      screen.blit(ck_img,[400,325])
```

```

42     screen.blit(ck_img,[720,325])
43     pg.display.update()
44
45     def init_bb_imgs() -> tuple[list[pg.Surface], list[int]]: # 爆弾拡大、加速機能
46         b_img = []
47         bb_accs = [a for a in range(1, 11)]
48         for r in range(1, 11):
49             bb_img = pg.Surface((20*r, 20*r))
50             pg.draw.circle(bb_img, (255, 0, 0), (10*r, 10*r), 10*r)
51             b_img.append(bb_img)
52         return b_img, bb_accs
53     """
54     def get_kk_img(sum_mv: tuple[int, int]) -> pg.Surface:
55         kk_dict = {
56             (0, -5): kk_img = pg.transform.rotozoom(pg.image.load("fig/3.png"), 270, 0.9),
57             (+5, -5): kk_img = pg.transform.rotozoom(pg.image.load("fig/3.png"), 315, 0.9),
58             (+5, 0): kk_img = pg.transform.rotozoom(pg.image.load("fig/3.png"), 0, 0.9),
59         }
60         if sum_mv == [0, -5]:
61             return
62     """
63
64     def main():
65         pg.display.set_caption("逃げろ！こうかとん")
66         screen = pg.display.set_mode((WIDTH, HEIGHT))
67
68         # こうかとん初期化
69         bg_img = pg.image.load("fig/campas.jpg")
70         scale=0.9
71         kk_base_img=pg.image.load("fig/3.png")
72         kk_img = pg.transform.rotozoom(kk_base_img, 0, scale)
73         kk_rct = kk_img.get_rect()
74         kk_rct.center = 550, 500
75         """
76         # 爆弾初期化
77         bb_imgs, bb_accs = init_bb_imgs()
78         bb_img = pg.Surface((20, 20))
79         pg.draw.circle(bb_img, (255, 0, 0), (10, 10), 10)
80         bb_rct = bb_img.get_rect()
81         bb_rct.center = random.randint(0, WIDTH), random.randint(0, HEIGHT)
82         bb_img.set_colorkey((0, 0, 0))
83         vx, vy = +5, +5 # 爆弾の速度
84         """
85         clock = pg.time.Clock()
86         tmr = 0
87         speed = 1
88         while True:
89             for event in pg.event.get():
90                 if event.type == pg.QUIT:
91                     return
92                 screen.blit(bg_img, [0, 0])
93
94                 #右上の時間表示count up (以下四文、消していいよ)
95                 font = pg.font.Font(None, 50)
96                 elapsed_time = tmr // 50
97                 time_text = font.render(f"Time: {elapsed_time}s", True, (255, 255, 255))
98                 screen.blit(time_text, (50, 30))
99

```

```
100         # if kk_rct.colliderect(bb_rct):
101         #     gameover(screen)
102         #     time.sleep(5)
103         #     return
104         # """
105         key_lst = pg.key.get_pressed()
106         sum_mv = [0, 0]
107         for key, mv in DELTA.items():
108             if key_lst[key]:
109                 sum_mv[0] += mv[0] # 左右方向
110                 sum_mv[1] += mv[1] # 上下方向
111         sum_mv[0] *= speed
112         sum_mv[1] *= speed
113
114         elapsed_sec = tmr // 50
115         scale = 0.9 + 0.2 * (elapsed_sec // 20) # 10秒ごとにサイズアップ
116         kk_img = pg.transform.rotozoom(kk_base_img, 0, scale)
117         kk_rct = kk_img.get_rect(center=kk_rct.center)
118
119
120         # obj_wall1 = pg.transform.rotozoom(obj_base_wall1, 0, scale)
121         # obj_wall2 = pg.transform.rotozoom(obj_base_wall2, 0, scale)
122         # obj_wall3 = pg.transform.rotozoom(obj_base_wall3, 0, scale)
123         # obj_wall1_rect = obj_wall1.get_rect(center=obj_wall1_rect.center)
124         # obj_wall2_rect = obj_wall2.get_rect(center=obj_wall2_rect.center)
125         # obj_wall3_rect = obj_wall3.get_rect(center=obj_wall3_rect.center)
126
127         speed = 1 + elapsed_sec // 10
128
129
130
131         kk_rct.move_ip(sum_mv) # こうかとの移動
132         if check_bound(kk_rct) != (True, True): # 画面外だったら
133             kk_rct.move_ip(-sum_mv[0], -sum_mv[1]) # 画面内に戻す
134         screen.blit(kk_img, kk_rct)
135
136
137
138
139
140         pg.display.update()
141         tmr += 1
142         clock.tick(50)
143
144
145     if __name__ == "__main__":
146         pg.init()
147         main()
148         pg.quit()
149         sys.exit()
150
151 # import os
152 # import random
153 # import sys
154 # import time
155 # import pygame as pg
156
157 # WIDTH = 650 # ゲームウィンドウの幅
```

```
158 # HEIGHT = 110 # ゲームウィンドウの高さ
159 # NUM_OF_BOMBS = 5 # 爆弾の個数
160 # os.chdir(os.path.dirname(os.path.abspath(__file__)))
161
162
163 # class Bird:
164 #     """
165 #     ゲームキャラクター（こうかとん）に関するクラス
166 #     """
167 #     delta = { # 押下キーと移動量の辞書
168 #         pg.K_UP: (0, -5),
169 #         pg.K_DOWN: (0, +5),
170 #         pg.K_LEFT: (-5, 0),
171 #         pg.K_RIGHT: (+5, 0),
172 #     }
173 #     img0 = pg.transform.rotozoom(pg.image.load("fig/campus.jpg"), 0, 0.9)
174 #     img = pg.transform.flip(img0, True, False) # デフォルトのこうかとん（右向き）
175 #     imgs = { # 0度から反時計回りに定義
176 #         (+5, 0): img, # 右
177 #         (+5, -5): pg.transform.rotozoom(img, 45, 0.9), # 右上
178 #         (0, -5): pg.transform.rotozoom(img, 90, 0.9), # 上
179 #         (-5, -5): pg.transform.rotozoom(img0, -45, 0.9), # 左上
180 #         (-5, 0): img0, # 左
181 #         (-5, +5): pg.transform.rotozoom(img0, 45, 0.9), # 左下
182 #         (0, +5): pg.transform.rotozoom(img, -90, 0.9), # 下
183 #         (+5, +5): pg.transform.rotozoom(img, -45, 0.9), # 右下
184 #     }
185
186 #     def __init__(self, xy: tuple[int, int]):
187 #         """
188 #         こうかとん画像Surfaceを生成する
189 #         引数 xy: こうかとん画像の初期位置座標タプル
190 #         """
191 #         self.img = __class__.imgs[(+5, 0)]
192 #         self.rct: pg.Rect = self.img.get_rect()
193 #         self.rct.center = xy
194
195 #     def change_img(self, num: int, screen: pg.Surface):
196 #         """
197 #         こうかとん画像を切り替え、画面に転送する
198 #         引数1 num: こうかとん画像ファイル名の番号
199 #         引数2 screen: 画面Surface
200 #         """
201 #         self.img = pg.transform.rotozoom(pg.image.load(f"fig/{num}.png"), 0, 0.9)
202 #         screen.blit(self.img, self.rct)
203
204 #     def update(self, key_lst: list[bool], screen: pg.Surface):
205 #         """
206 #         押下キーに応じてこうかとんを移動させる
207 #         引数1 key_lst: 押下キーの真値リスト
208 #         引数2 screen: 画面Surface
209 #         """
210 #         sum_mv = [0, 0]
211 #         for k, mv in __class__.delta.items():
212 #             if key_lst[k]:
213 #                 sum_mv[0] += mv[0]
214 #                 sum_mv[1] += mv[1]
215 #         self.rct.move_ip(sum_mv)
```

```
216     #         if check_bound(self.rct) != (True, True):
217     #             self.rct.move_ip(-sum_mv[0], -sum_mv[1])
218     #         if not (sum_mv[0] == 0 and sum_mv[1] == 0):
219     #             self.img = __class__.imgs[tuple(sum_mv)]
220     #         screen.blit(self.img, self.rct)
221
222
223     # def main():
224     #     pg.display.set_caption("たたかえ！こうかとん")
225     #     screen = pg.display.set_mode((WIDTH, HEIGHT))
226     #     bg_img = pg.image.load("fig/campus.jpg")
227     #     bird = Bird((300, 200))
228     #     tmr = 0
229     #     while True:
230
231
232     #         pg.display.update()
233
234
235     # if __name__ == "__main__":
236     #     pg.init()
237     #     main()
238     #     pg.quit()
239     #     sys.exit()
```