

C0B22120 / ProjExd05Public

<> CodeIssuesPull requestsActionsProjectsWikiSecurityInsightsSettings

main

ProjExd05 / README.md

Go to file

t...

C0B22120 修正3 調整,回復

now...

24 lines (22 loc) · 744 Bytes

PreviewCodeBlame

RawCopyDownloadEditDropdownMenu

# ゲームのタイトル

---

## 実行環境の必要条件

---

- python >= 3.10
- pygame >= 2.1

## ゲームの概要

---

主人公キャラクターをキー操作で動かし襲い掛かる敵からタワーを守るゲーム。

## ゲームの実装

---

### 共通基本機能

- 主人公キャラクターに関するクラス
- 敵の呼び出しに関するクラス
- タワーに関するクラス

### 担当追加機能

- UI全般に関する改修
- tower回復(右シフト)

### ToDo

- 変数の整理

### メモ

- 画像名称は分かりやすいように整理済
- 画像サイズは間隔で処理しているため微修正の余地あり
- 回復は不要ではある

## 参照

- 画像 いらすとや (<https://www.irasutoya.com/>)

C0B22120 / ProjExd05 Public

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

main ProjExd05 / tower\_defence.py

Go to file

t

...



C0B22120 修正3 調整,回復

1 minute ago



245 lines (212 loc) · 8.57 KB

Code

Blame

Raw



```
1  import math
2  import random
3  import sys
4  import time
5  from typing import Any
6
7  import pygame as pg
8  from pygame.sprite import AbstractGroup
9
10
11  WIDTH = 1600 # ゲームウィンドウの幅
12  HEIGHT = 900 # ゲームウィンドウの高さ
13
14  def check_bound(obj: pg.Rect) -> tuple[bool, bool]:
15      """
16      オブジェクトが画面内か画面外かを判定し、真理値タプルを返す
17      引数 obj: オブジェクト (爆弾, こうかとん, ビーム) SurfaceのRect
18      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)
19      """
20      yoko, tate = True, True
21      if obj.left < 0 or WIDTH < obj.right: # 横方向のはみ出し判定
22          yoko = False
23      if obj.top < 0 or HEIGHT < obj.bottom: # 縦方向のはみ出し判定
24          tate = False
25      return yoko, tate
26
27  def calc_orientation(org: pg.Rect, dst: pg.Rect) -> tuple[float, float]:
28      """
29      orgから見て、dstがどこにあるかを計算し、方向ベクトルをタプルで返す
30      引数1 org: 爆弾SurfaceのRect
31      引数2 dst: こうかとんSurfaceのRect
32      戻り値: orgから見たdstの方向ベクトルを表すタプル
33      """
34      x_diff, y_diff = dst.centerx-org.centerx, dst.centery-org.centery
35      norm = math.sqrt(x_diff**2+y_diff**2)
36      return x_diff/norm, y_diff/norm
37
38  class Hero(pg.sprite.Sprite):
39      """
40      ゲームキャラクター (こうかとん) に関するクラス
41      """
42      delta = { # 押下キーと移動量の辞書
43          pg.K_UP: (0, -1),
44          pg.K_DOWN: (0, +1),
```

```

45         pg.K_LEFT: (-1, 0),
46         pg.K_RIGHT: (+1, 0),
47     }
48
49     def __init__(self, xy: tuple[int, int]):
50         """
51         こうかとん画像Surfaceを生成する
52         引数1 num: こうかとん画像ファイル名の番号
53         引数2 xy: こうかとん画像の位置座標タプル
54         """
55         super().__init__()
56         img0 = pg.transform.rotozoom(pg.image.load(f"ex05/fig/hero.png"), 0, 0.1)
57         img = pg.transform.flip(img0, True, False) # デフォルトのこうかとん
58         self.imgs = {
59             (+1, 0): img, # 右
60             (+1, -1): pg.transform.rotozoom(img, 45, 1.0), # 右上
61             (0, -1): pg.transform.rotozoom(img, 90, 1.0), # 上
62             (-1, -1): pg.transform.rotozoom(img0, -45, 1.0), # 左上
63             (-1, 0): img0, # 左
64             (-1, +1): pg.transform.rotozoom(img0, 45, 1.0), # 左下
65             (0, +1): pg.transform.rotozoom(img, -90, 1.0), # 下
66             (+1, +1): pg.transform.rotozoom(img, -45, 1.0), # 右下
67         }
68         self.dire = (+1, 0)
69         self.image = self.imgs[self.dire]
70         self.rect = self.image.get_rect()
71         self.rect.center = xy
72         self.speed = 10
73
74     def change_img(self, num: str, screen: pg.Surface):
75         """
76         主人公画像を切り替え、画面に転送する
77         引数1 num: 変更後画像ファイル名
78         引数2 screen: 画面Surface
79         """
80         self.image = pg.transform.rotozoom(pg.image.load(f"ex05/fig/{num}.png"), 0, 0.1)
81         screen.blit(self.image, self.rect)
82
83     def update(self, key_lst: list[bool], screen: pg.Surface):
84         """
85         押下キーに応じて主人公を移動させる
86         引数1 key_lst: 押下キーの真理値リスト
87         引数2 screen: 画面Surface
88         """
89         sum_mv = [0, 0]
90         for k, mv in __class__.delta.items():
91             if key_lst[k]:
92                 self.rect.move_ip(+self.speed*mv[0], +self.speed*mv[1])
93                 sum_mv[0] += mv[0]
94                 sum_mv[1] += mv[1]
95         if check_bound(self.rect) != (True, True):
96             for k, mv in __class__.delta.items():
97                 if key_lst[k]:
98                     self.rect.move_ip(-self.speed*mv[0], -self.speed*mv[1])
99         if not (sum_mv[0] == 0 and sum_mv[1] == 0):
100             self.dire = tuple(sum_mv)
101             self.image = self.imgs[self.dire]
102             screen.blit(self.image, self.rect)
103
104     def get_direction(self) -> tuple[int, int]:

```

```

105         return self.dire
106
107
108 class Tower(pg.sprite.Sprite):
109     """
110     守るタワーについて設定するクラス
111
112     """
113     def __init__(self) :
114         super().__init__()
115         self.image=pg.transform.rotozoom(pg.image.load(f"ex05/fig/tower.png"), 0, 0.8)
116         self.life = 3
117         self.rect = self.image.get_rect()
118         self.rect.center = WIDTH/2,HEIGHT/2
119
120         self.font = pg.font.Font(None, 50) #life表示のための用意
121         self.color = (255, 0 , 0)
122         self.displife = self.font.render(f"Life: {self.life}", 0, self.color)
123
124
125     def update(self,screen):
126         """
127         現在のlifeやタワーを表示する
128         """
129         screen.blit(self.image,self.rect)
130         self.displife = self.font.render(f"Life: {self.life}", 0, self.color)
131         screen.blit(self.displife,(30,HEIGHT-125))
132
133 class Enemy(pg.sprite.Sprite):
134     """
135     敵機に関するクラス
136     """
137     imgs = [pg.transform.rotozoom(pg.image.load(f"ex05/fig/ene{i}.png"),0,0.15) for i in range(1, 4)]
138
139     def __init__(self,place: int,tower: Tower):
140         super().__init__()
141         self.image = random.choice(__class__.imgs)
142         self.rect = self.image.get_rect()
143         if place == 0: #以下のif文で呼び出し時の値によって出現場所を決める
144             self.rect.center = random.randint(0, WIDTH), 0
145         elif place == 1:
146             self.rect.center = WIDTH,random.randint(0,HEIGHT)
147         elif place == 2:
148             self.rect.center = random.randint(0,WIDTH),HEIGHT
149         elif place == 3:
150             self.rect.center = 0,random.randint(0,HEIGHT)
151         self.vx, self.vy = calc_orientation(self.rect, tower.rect) #towerに向かうように方向ベクトルの設定
152         self.speed = 6
153
154     def update(self):
155         """
156         敵機を速度ベクトルself.vx,vyに基づき移動させる
157         引数 screen : 画面Surface
158         """
159         self.rect.move_ip(+self.speed*self.vx, +self.speed*self.vy)
160
161
162
163 class Score:
164     """

```

```

165     打ち落とした敵機の数スコアとして表示するクラス
166     敵機：1点
167     """
168     def __init__(self):
169         self.font = pg.font.Font(None, 50)
170         self.color = (0, 0, 0)
171         self.score = 0
172         self.image = self.font.render(f"Score: {self.score}", 0, self.color)
173         self.rect = self.image.get_rect()
174         self.rect.center = 100, HEIGHT-50
175
176     def score_up(self, add):
177         self.score += add
178
179     def update(self, screen: pg.Surface):
180         self.image = self.font.render(f"Score: {self.score}", 0, self.color)
181         screen.blit(self.image, self.rect)
182
183     def main():
184         pg.display.set_caption("守れ!こうかとん")
185         screen = pg.display.set_mode((WIDTH, HEIGHT))
186         bg_img = pg.image.load("ex05/fig/bg_natural_mori.jpg")
187
188         score = Score()
189         hero = Hero( (900, 400))
190         emys = pg.sprite.Group()
191         tower = Tower()
192         fonto = pg.font.Font(None, 50)
193         tmr = 0
194         clock = pg.time.Clock()
195
196         while True:
197             key_lst = pg.key.get_pressed()
198             for event in pg.event.get():
199                 if event.type == pg.QUIT:
200                     return 0
201                 if event.type == pg.KEYDOWN and event.key == pg.K_RSHIFT:
202                     if score.score >= 50:
203                         tower.life += 1
204                         score.score -= 50
205
206             screen.blit(bg_img, [0, 0])
207
208             if tmr%20 == 0: # 200フレームに1回、敵機を出現させる
209                 emys.add(Enemy(random.randint(0,3),tower)) #randintで出現位置を四方向から選び、towerで方向の指定
210             for emy in pg.sprite.spritecollide(hero, emys, True):
211                 emy.kill()
212                 score.score_up(1)
213
214             for emy in pg.sprite.spritecollide(tower, emys, True):
215                 if tower.life >= 2:
216                     tower.life -=1 #lifeを減らす
217                     emy.kill()
218                 else:
219                     screen.blit(pg.transform.rotozoom(pg.image.load("ex05/fig/text_gameover.png"),0,0.4),[600,250])
220                     hero.change_img("lose", screen) # 悲しみエフェクト
221                     tower.life -=1
222                     score.update(screen)
223                     tower.update(screen)
224                     screen.blit(txt_time, [WIDTH/2, 50])
225
226         pg.quit()

```

```
225         pg.display.update()
226         time.sleep(2)
227         return
228
229     txt_time = fonto.render(str(tmr/50), True, (0, 0, 0))
230     screen.blit(txt_time, [WIDTH/2, 50])
231     hero.update(key_lst, screen)
232     emys.update()
233     emys.draw(screen)
234     score.update(screen)
235     tower.update(screen)
236     pg.display.update()
237     tmr += 1
238     clock.tick(50)
239
240
241 if __name__ == "__main__":
242     pg.init()
243     main()
244     pg.quit()
245     sys.exit()
```