


 c0b2304920 /
ProjExD_Group08



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

ProjExD_Group08 / README.md 

...



c0a23059fb readme微修正

88cb3cb · 6 hours ago



42 lines (35 loc) · 1.44 KB

Preview Code Blame

Raw    

ボンバーこうかтон

実行環境の必要条件

- python >= 3.10
- pygame >= 2.1

ゲームの概要

- 障害物も敵も爆弾で破壊するゲーム with こうかтон
- 参考URL : [レトロゲームの殿堂](#)

ゲームの遊び方

- 矢印キーでボンバーKOKAを操作し、スペースキー押下による足元への爆弾設置。
- 三分以内に他キャラクターを殲滅、もしくは最大スコアの状態で勝利。

ゲームの実装

共通基本機能

- 主人公キャラ
- 矢印操作機能
- 盤面領域内判定関数

分担追加機能

- Helo(操作キャラ)クラス(北村)
- 壁の生成、破壊機能クラス(北村)
- マップ詳細生成機能クラス(北村)
- 敵クラス(小林)

- 爆弾の制御機能クラス(小林)
- 他キャラの行動機能クラス(小林)
- タイマー機能クラス (町田)
- 効果音、BGM制御機能クラス(町田)
- スコア機能クラス(おん)
- タイトル画面描画機能クラス(岡崎)
- ゲームオーバー画面描画クラス(岡崎)

ToDo

- [1] マージ後のコメントなどの精査、修正
- [2] 記述方法の統一
- [3] 素材作成
- [4] 細かな機能の発案、実装

メモ

- main関数は最低限の呼び出しのみで記述している
- 汎用的な関数を用意して、動作対象に適用している

c0b2304920 /
ProjExD_Group08

<> Code

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

ProjExD_Group08 / bomber_kokaton.py



c0b2304920 ブロックの判定 実装

951b0bd · 4 days ago



124 lines (106 loc) · 4.11 KB

Code

Blame

Raw



```
1  import os
2  import random
3  import sys
4  import time
5
6  import pygame as pg
7
8
9  WIDTH, HEIGHT = 750, 700
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 < 50 or WIDTH - 50 < obj_rct.right: # ブロックにぶつかったら止まるように
21          yoko = False
22      if obj_rct.top < 100 or HEIGHT - 50 < obj_rct.bottom:
23          tate = False
24      for i in range(6):
25          num = 100*i
26          if (100 + num) < obj_rct.left < (150 + num) or (100 + num) < obj_rct.right < (150 + num):
27              for j in range(5):
28                  num = 100 * j
29                  if 150 + num < obj_rct.top < 200 + num or 150 + num < obj_rct.bottom < 200 + num:
30                      yoko = False
31                      tate = False
32      return yoko, tate
33
34
35  class Hero:
36      """
37      ゲームキャラクター（こうかとん）に関するクラス
38      """
39      delta = { # 押下キーと移動量の辞書
40          pg.K_UP: (0, -5),
41          pg.K_DOWN: (0, +5),
42          pg.K_LEFT: (-5, 0),
43          pg.K_RIGHT: (5, 0),
44      }
```

```

43     pg.K_RIGHT: (+5, 0),
44 }
45 img0 = pg.transform.rotozoom(pg.image.load("images/3.png"), 0, 0.9)
46 img = pg.transform.flip(img0, True, False) # デフォルトのこうかとん (右向き)
47 imgs = { # 0度から反時計回りに定義
48     (+5, 0): img, # 右
49     (+5, -5): pg.transform.rotozoom(img, 45, 0.9), # 右上
50     (0, -5): pg.transform.rotozoom(img, 90, 0.9), # 上
51     (-5, -5): pg.transform.rotozoom(img0, -45, 0.9), # 左上
52     (-5, 0): img0, # 左
53     (-5, +5): pg.transform.rotozoom(img0, 45, 0.9), # 左下
54     (0, +5): pg.transform.rotozoom(img, -90, 0.9), # 下
55     (+5, +5): pg.transform.rotozoom(img, -45, 0.9), # 右下
56 }
57
58 def __init__(self, xy: tuple[int, int]):
59     """
60     こうかとん画像Surfaceを生成する
61     引数 xy: こうかとん画像の初期位置座標タプル
62     """
63     self.img = __class__.imgs[(+5, 0)]
64     self.rct: pg.Rect = self.img.get_rect()
65     self.rct.center = xy
66     self.dire = (+5, 0) # 演習3
67
68 def change_img(self, num: int, screen: pg.Surface):
69     """
70     こうかとん画像を切り替え、画面に転送する
71     引数1 num: こうかとん画像ファイル名の番号
72     引数2 screen: 画面Surface
73     """
74     self.img = pg.transform.rotozoom(pg.image.load(f"fig/{num}.png"), 0, 0.9)
75     screen.blit(self.img, self.rct)
76
77 def update(self, key_lst: list[bool], screen: pg.Surface):
78     """
79     押下キーに応じてこうかとんを移動させる
80     引数1 key_lst: 押下キーの真値リスト
81     引数2 screen: 画面Surface
82     """
83     sum_mv = [0, 0]
84     for k, mv in __class__.delta.items():
85         if key_lst[k]:
86             sum_mv[0] += mv[0]
87             sum_mv[1] += mv[1]
88     self.rct.move_ip(sum_mv)
89     if check_bound(self.rct) != (True, True):
90         self.rct.move_ip(-sum_mv[0], -sum_mv[1])
91     if not (sum_mv[0] == 0 and sum_mv[1] == 0):
92         self.img = __class__.imgs[tuple(sum_mv)]
93         self.dire = sum_mv # 更新
94     screen.blit(self.img, self.rct)
95
96
97 def main():
98     pg.display.set_caption("ボンバーこうかとん")
99     screen = pg.display.set_mode((WIDTH, HEIGHT))
100     bg_img = pg.image.load("images/bg_ver.1.0.png") # 一時的な背景(緑スペース背景)
101     hero = Hero((75, 125))

```

```
101     hero = hero((0, 100))
102     clock = pg.time.Clock()
103     tmr = 0
104
105     while True:
106         for event in pg.event.get():
107             if event.type == pg.QUIT:
108                 return
109
110         screen.blit(bg_img, [0, 50])
111
112         key_lst = pg.key.get_pressed()
113         hero.update(key_lst, screen)
114
115         pg.display.update()
116         tmr += 1
117         clock.tick(50)
118
119
120 if __name__ == "__main__":
121     pg.init()
122     main()
123     pg.quit()
124     sys.exit()
```