




 c0a2302825 / ProjExD_Group13




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

 allatack ▾

ProjExD_Group13 / Shouting_game.py



 t ...



c0a2302825 全方位攻撃

866d5cc · 1 hour ago



293 lines (251 loc) · 10.6 KB

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

[Raw](#)     

```
1 import pygame as pg
2 import random
3 import math
4
5 pg.init()
6
7 # ディスプレイの設定
8 SCREEN_WIDTH = pg.display.Info().current_w
9 SCREEN_HEIGHT = pg.display.Info().current_h
10 screen = pg.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT), pg.FULLSCREEN)
11 pg.display.set_caption("しゅうていんぐげえむ")
12
13 # 色の定義
14 WHITE = (255, 255, 255)
15 RED = (255, 0, 0)
16 GREEN = (0, 255, 0)
17 BLUE = (0, 0, 255)
18
19 # ゲームエリア
20 GAME_AREA_SIZE = min(SCREEN_WIDTH, SCREEN_HEIGHT) * 0.6
21 GAME_AREA_X = (SCREEN_WIDTH - GAME_AREA_SIZE) / 2
22 GAME_AREA_Y = (SCREEN_HEIGHT - GAME_AREA_SIZE) / 2
23
24
25 class Player:
26     """
27     Playerの操作するキャラのクラス
28     """
29     def __init__(self):
30         self.width = 50
31         self.height = 50
32         self.x = SCREEN_WIDTH // 2 - self.width // 2
33         self.y = SCREEN_HEIGHT // 2 - self.height // 2
34         self.speed = 5
35         self.hp = 100 # プレイヤーHPの追加 (初期化)
36         self.sp = 0 # プレイヤーSP(スキルポイント)の追加 (初期化)
37
38     def move(self, dx:int, dy:int):
39         """
40         自機を速度ベクトルself.x,self.yに基づき、
41         new_x,new_yとして移動させる
```

```
42     プレイヤーの行動範囲を制御する
43     """
44     new_x = self.x + dx * self.speed
45     new_y = self.y + dy * self.speed
46     if (GAME_AREA_X < new_x < GAME_AREA_X + GAME_AREA_SIZE - self.width and
47         GAME_AREA_Y < new_y < GAME_AREA_Y + GAME_AREA_SIZE - self.height):
48         self.x = new_x
49         self.y = new_y
50
51     def draw(self, screen: pg.Surface):
52         """
53         引数 screen : 画面surface
54         """
55         pg.draw.rect(screen, BLUE, (self.x, self.y, self.width, self.height))
56
57
58     class Enemy:
59         """
60         敵キャラを表示するクラス
61         """
62         def __init__(self):
63             self.width = 60
64             self.height = 60
65             self.x = random.randint(0, SCREEN_WIDTH - self.width)
66             self.y = 50
67             self.speed = random.uniform(2, 5)
68             self.hp = 100 # 敵HPの追加 (初期化)
69             self.direction = random.choice([-1, 1])
70             self.change_direction_counter = 0 # 敵の移動判定のカウンターの初期化
71             self.change_direction_threshold = random.randint(60, 180) # 敵の停止時間をランダム値で設定
72
73         def move(self):
74             """
75             敵キャラを速度ベクトルself.x,self.directionに基づき移動させる
76             """
77             self.x += self.speed * self.direction
78             if self.x <= 0 or self.x >= SCREEN_WIDTH - self.width:
79                 self.direction *= -1
80
81             self.change_direction_counter += 1 # 敵の移動判定のカウンターの更新
82             if self.change_direction_counter >= self.change_direction_threshold: # 敵の停止時間を超えたら敵:
83                 self.direction = random.choice([-1, 1]) # 敵の動くy軸+-の方向をランダムに設定
84                 self.speed = random.uniform(2, 5) # 敵の移動量をランダムに設定
85                 self.change_direction_counter = 0 # 敵の移動判定のカウンターのリセット
86                 self.change_direction_threshold = random.randint(60, 180) # 敵の停止時間をランダム値で設定
87
88         def draw(self, screen: pg.Surface):
89             """
90             引数 screen : 画面surface
91             """
92             pg.draw.rect(screen, RED, (self.x, self.y, self.width, self.height))
93
94
95     class Bullet:
96         """
97         敵味方が攻撃を行う弾を表すクラス。
98
99         変数:
```

```
100         x : 弾の現在のx座標
101         y : 弾の現在のy座標
102         dx : x方向の移動速度
103         dy : y方向の移動速度
104
105     メソッド:
106         move(): 弾を移動させる
107         draw(screen): 弾を画面上に描画する
108     """
109     def __init__(self, x:float, y:float, target_x:float, target_y:float):
110         """
111         Bulletオブジェクトを初期化する。
112
113         引数:
114             x : 弾の初期x座標
115             y : 弾の初期y座標
116             target_x : プレイヤーのx座標
117             target_y : プレイヤーのy座標
118         """
119         self.x = x
120         self.y = y
121         angle = math.atan2(target_y - y, target_x - x)
122         speed = 5
123         self.dx = math.cos(angle) * speed
124         self.dy = math.sin(angle) * speed
125
126     def move(self):
127         """弾を現在の速度に基づいて移動させる。"""
128         self.x += self.dx
129         self.y += self.dy
130
131     def draw(self, screen: pg.Surface):
132         """
133         弾を画面上に描画する。
134
135         引数:
136             screen (pygame.Surface): 描画対象の画面
137         """
138         pg.draw.circle(screen, WHITE, (int(self.x), int(self.y)), 5)
139
140
141     class OmniBullet:
142         """
143         プレイヤーが全方向に発射する弾のクラス。
144         属性:
145             bullets (list): 発射された弾を保持するリスト。
146         メソッド:
147             __init__(x, y): OmniBulletオブジェクトを初期化する。
148             move(): 全ての弾を移動させる。
149             draw(screen): 全ての弾を画面上に描画する。
150         """
151     def __init__(self, x, y):
152         """
153         OmniBulletオブジェクトを初期化する。
154         引数:
155             x (float): 弾の初期x座標。
156             y (float): 弾の初期y座標。
157         """
```

```
158         self.bullets = []
159         speed = 5
160         for angle in range(0, 360, 45): # 45度間隔で全方向に弾を作成
161             radians = math.radians(angle)
162             dx = math.cos(radians) * speed
163             dy = math.sin(radians) * speed
164             self.bullets.append(Bullet(x, y, x + dx * 10, y + dy * 10))
165     def move(self):
166         """
167         全ての弾を現在の速度に基づいて移動させる。
168         """
169         for bullet in self.bullets:
170             bullet.move()
171     def draw(self, screen):
172         """
173         全ての弾を画面上に描画する。
174         引数:
175             screen (pygame.Surface): 描画対象の画面。
176         """
177         for bullet in self.bullets:
178             bullet.draw(screen)
179
180
181     def main():
182         player = Player()
183         enemy = Enemy() # enemy関数の呼び出し
184         player_bullets = [] # プレイヤーと敵の弾を保持するリスト
185         enemy_bullets = []
186         clock = pg.time.Clock()
187         omni_bullets = [] # 全方向攻撃の弾を保持するリスト
188
189         running = True
190         while running:
191             for event in pg.event.get():
192                 if event.type == pg.QUIT:
193                     running = False
194                 elif event.type == pg.KEYDOWN:
195                     if event.key == pg.K_ESCAPE:
196                         running = False
197                     elif event.key == pg.K_SPACE: # スペースキーで弾の発射
198                         player_bullets.append(Bullet(player.x + player.width // 2, player.y,
199                                                         player.x + player.width // 2, 0))
200                     elif event.key == pg.K_z: # Zキーで全方向攻撃
201                         if player.sp >= 5: # SPゲージが5以上の場合
202                             omni_bullets.append(OmniBullet(player.x + player.width // 2, player.y + play
203                                                                 player.sp -= 5 # SPゲージ5を消費して全方位攻撃
204
205
206             keys = pg.key.get_pressed()
207             player.move(keys[pg.K_RIGHT] - keys[pg.K_LEFT], keys[pg.K_DOWN] - keys[pg.K_UP])
208
209             enemy.move()
210
211             if random.random() < 0.02: # 弾の発生
212                 # 画面の四辺からランダムに弾を発射
213                 side = random.choice(['top', 'bottom', 'left', 'right'])
214                 if side == 'top':
215                     x = random.randint(0, SCREEN_WIDTH)
```

```
126         y = 0
127     elif side == 'bottom':
128         x = random.randint(0, SCREEN_WIDTH)
129         y = SCREEN_HEIGHT
130     elif side == 'left':
131         x = 0
132         y = random.randint(0, SCREEN_HEIGHT)
133     else: # right
134         x = SCREEN_WIDTH
135         y = random.randint(0, SCREEN_HEIGHT)
136
137     target_x = GAME_AREA_X + GAME_AREA_SIZE // 2
138     target_y = GAME_AREA_Y + GAME_AREA_SIZE // 2
139     enemy_bullets.append(Bullet(x, y, target_x, target_y))
140
141 # 全方向攻撃の弾の移動と当たり判定
142 for omni in omni_bullets[:]:
143     for bullet in omni.bullets[:]:
144         bullet.move()
145         if (bullet.x < 0 or bullet.x > SCREEN_WIDTH or
146             bullet.y < 0 or bullet.y > SCREEN_HEIGHT):
147             omni.bullets.remove(bullet)
148         elif (enemy.x < bullet.x < enemy.x + enemy.width and
149             enemy.y < bullet.y < enemy.y + enemy.height):
150             enemy.hp -= 10 # 敵HPの更新
151             omni.bullets.remove(bullet)
152     if not omni.bullets:
153         omni_bullets.remove(omni)
154
155 # プレイヤーの弾の移動と当たり判定
156 for bullet in player_bullets[:]: # 弾の動きと衝突
157     bullet.move()
158     if bullet.y < 0:
159         player_bullets.remove(bullet)
160     elif (enemy.x < bullet.x < enemy.x + enemy.width and
161         enemy.y < bullet.y < enemy.y + enemy.height):
162         enemy.hp -= 10 # 敵HPの更新
163         player.sp += 5 # プレイヤーSPの更新
164         player_bullets.remove(bullet)
165
166 # 敵の弾の移動と当たり判定
167 for bullet in enemy_bullets[:]:
168     bullet.move()
169     if (bullet.x < 0 or bullet.x > SCREEN_WIDTH or
170         bullet.y < 0 or bullet.y > SCREEN_HEIGHT):
171         enemy_bullets.remove(bullet)
172     elif (player.x < bullet.x < player.x + player.width and
173         player.y < bullet.y < player.y + player.height):
174         player.hp -= 1 # プレイヤーHPの更新
175         enemy_bullets.remove(bullet)
176
177 if player.hp <= 0 or enemy.hp <= 0: # ゲームの終了判定
178     running = False # ゲームを終了させる
179
180 screen.fill((0, 0, 0))
181 # プレイヤーの行動範囲を視覚的に表示する
182 pg.draw.rect(screen, WHITE, (GAME_AREA_X, GAME_AREA_Y, GAME_AREA_SIZE, GAME_AREA_SIZE), 2)
183 player.draw(screen)
```

```
274     # 敵キャラを表示
275     enemy.draw(screen)
276     for bullet in player_bullets + enemy_bullets: # 弾の描画
277         bullet.draw(screen)
278     for omni in omni_bullets:
279         omni.draw(screen)
280
281
282     pg.draw.rect(screen, RED, (10, SCREEN_HEIGHT - 30, player.hp * 2, 20)) # プレイヤーHPのゲージ
283     pg.draw.rect(screen, GREEN, (10, 10, enemy.hp * 2, 20)) # 敵HPのゲージを表示
284     pg.draw.rect(screen, BLUE, (SCREEN_WIDTH - 210, SCREEN_HEIGHT - 30, player.sp * 2, 20)) # フ
285
286     pg.display.flip()
287     clock.tick(60)
288
289     pg.quit()
290
291
292 if __name__ == "__main__":
293     main()
```