c0a22043 BGM機能 4 minutes ago

289 lines (263 loc) · 10.3 KB

Code Blame Raw

```python
import pygame
import sys
import random
import numpy as np
import datetime

OCTO_CAT_VELOCITY = 4
OCTO_CAT_JUMP = 20
COLORS = [[255,0,0],[255,165,0],[255,255,0],[0,255,0],[0,255,255],[0,0,255],[128,0,128]]

pygame.init()
screen = pygame.display.set_mode((640, 480))
pygame.display.set_caption("Jump the Rope")
heart_image = pygame.image.load("ex05/images/heart.png")
clock = pygame.time.Clock()

class Octo_Cat:
    def __init__(self,x,y):
        #position
        self.x = x
        self.y = y
        #image before scaling
        self.rough_image = pygame.image.load("ex05/images/greenoctocat.png").convert()
        #properly scaled image
        self.image = pygame.transform.scale(self.rough_image, (20,20))
        #image to show when the player is jumping
        self.immune_image = pygame.transform.scale(self.rough_image, (30,30))
        #about movements
        self.move_right = False
        self.move_left = False
        self.move_up = False
        self.move_down = False
        #if jumping or not
        self.immunity = False
        #the player cannot jump more than a specified amount of time
        self.immunity_count = 0
```

```
            self.life = 4
            #if the player is inflicted with damage, it cannot be inflicted again for a certain amou
            self.life_lost_time = 0

        #movements
        def update(self,event):
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT:
                    self.move_right = True
                if event.key == pygame.K_LEFT:
                    self.move_left = True
                if event.key == pygame.K_UP:
                    self.move_up = True
                if event.key == pygame.K_DOWN:
                    self.move_down = True
                if event.key == pygame.K_SPACE:
                    self.immunity = True
            elif event.type == pygame.KEYUP:
                if event.key == pygame.K_RIGHT:
                    self.move_right = False
                if event.key == pygame.K_LEFT:
                    self.move_left = False
                if event.key == pygame.K_UP:
                    self.move_up = False
                if event.key == pygame.K_DOWN:
                    self.move_down = False

    #defiens ropes and dots
    class Rope:
        def __init__(self,x=0,y=0,velocity=0,tilt=0,color=0):
            self.x = x
            self.y = y
            self.velocity = velocity
            self.tilt = tilt
            self.color = color
        def update(self):
            return
        def judge(self,octo_cat):
            return

    #vertical ropes
    class Straight_Rope(Rope):
        def update(self):
            if(self.x > 635):
                self.direction = "LEFT"
            elif(self.x < 5):
                self.direction = "RIGHT"
            if(self.direction == "RIGHT"):
                self.x += self.velocity
            elif(self.direction == "LEFT"):
                self.x -= self.velocity
            pygame.draw.line(screen, COLORS[3], [self.x, 0], [self.x, 480], 5)

        #checks if the player and the rope collided
        def judge(self,octo_cat):
```

```
            if(self.x > (octo_cat.x) and self.x < (octo_cat.x + 20)):
                return True
            else:
                return False

    #horizontal ropes
    class Straight_Rope_Horizontal(Rope):
        def update(self):
            if(self.y > 475):
                self.direction = "DOWN"
            elif(self.y < 5):
                self.direction = "UP"
            if(self.direction == "DOWN"):
                self.y -= self.velocity
            elif(self.direction == "UP"):
                self.y += self.velocity
            pygame.draw.line(screen, COLORS[3],[0, self.y], [640, self.y], 5)

        #checks if the player and the rope collided
        def judge(self,octo_cat):
            if(self.y > (octo_cat.y) and self.y < (octo_cat.y + 20)):
                return True
            else:
                return False

    #dots
    class Shooting_Star(Rope):
        def update(self):
            self.x += self.tilt
            self.y += self.velocity
            pygame.draw.circle(screen, COLORS[self.color], [self.x, self.y], 6)

        #checks if the player and the dot collided
        def judge(self,octo_cat):
            if((self.y > (octo_cat.y) and self.y < (octo_cat.y + 20)) and (self.x > (octo_cat.x) and
                return True
            else:
                return False

    def open():
        endFlag = False
        font1 = pygame.font.SysFont(None, 80)
        text1 = font1.render("Jump the Rope", False, (255,255,255))
        font2 = pygame.font.SysFont(None, 40)
        text2 = font1.render("Press Any Key to Start", False, (255,255,255))

        while endFlag == False:
            screen.fill((0,0,0))
            screen.blit(text1,(30,50))
            screen.blit(text2,(20,150))
            pygame.display.update()
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    endFlag = True
                elif event.type == pygame.KEYDOWN:
                    endFlag = True
```

```
149                     main()
150
151 def main():
152         global is_new_bgm_playing
153         endFlag = False
154         octo_cat = Octo_Cat(400,400)
155         time_elapsed = 0
156         force_quit = False
157         BGM = pygame.mixer.Sound('ex05/fig/BGM.mp3')
158         BGM.set_volume(0.2)  # 音量を設定（0.0から1.0の範囲）
159         BGM.play(-1)  # -1を渡すと無限ループ再生
160         New_BGM = pygame.mixer.Sound('ex05/fig/maou.mp3')
161         is_new_bgm_playing = False  # 新しいBGMの再生状態をトラッキング
162         game_over_sound = pygame.mixer.Sound('ex05/fig/dead.wav')
163
164
165         ropes = []
166
167         while endFlag == False:
168             clock.tick(60)
169             time_elapsed += 1
170             screen.fill((0,0,0))
171
172             for event in pygame.event.get():
173                 if event.type == pygame.QUIT:
174                     endFlag = True
175                     force_quit = True
176                 elif event.type == pygame.KEYDOWN:
177                     if event.key == pygame.K_TAB:
178                         if pygame.mixer.get_busy():
179                             BGM.stop()  # 現在のBGMを停止
180                             New_BGM.stop()
181                             New_BGM.play(-1)  # 新しいBGMを再生
182                     octo_cat.update(event)
183
184             #move the player
185             if octo_cat.move_right == True:
186                 if octo_cat.x < 620:
187                     octo_cat.x += OCTO_CAT_VELOCITY
188             if octo_cat.move_left == True:
189                 if octo_cat.x > 00:
190                     octo_cat.x -= OCTO_CAT_VELOCITY
191             if octo_cat.move_up == True:
192                 if octo_cat.y > 00:
193                     octo_cat.y -= OCTO_CAT_VELOCITY
194             if octo_cat.move_down == True:
195                 if octo_cat.y < 460:
196                     octo_cat.y += OCTO_CAT_VELOCITY
197
198             #make the instances of ropes and dots
199             if (time_elapsed == 20):
200                 straight_rope = Straight_Rope(0,0,3)
201                 ropes.append(straight_rope)
202             if (time_elapsed == 300):
203                 straight_rope_horizontal = Straight_Rope_Horizontal(0,0,3)
```

```
204                 ropes.append(straight_rope_horizontal)
205             if (time_elapsed == 600):
206                 straight_rope = Straight_Rope(0,0,3)
207                 ropes.append(straight_rope)
208             if (time_elapsed == 860):
209                 straight_rope_horizontal = Straight_Rope_Horizontal(0,0,3)
210                 ropes.append(straight_rope_horizontal)
211             if(random.randrange(200) < 6):
212                 shooting_star1 = Shooting_Star(random.randrange(640),0,random.randrange(5) + 5,rando
213                 ropes.append(shooting_star1)
214                 shooting_star2 = Shooting_Star(random.randrange(640),0,random.randrange(5) + 5,rando
215                 ropes.append(shooting_star2)
216                 shooting_star3 = Shooting_Star(10,random.randrange(480),random.randrange(5) + 5,rand
217                 ropes.append(shooting_star3)
218                 shooting_star4 = Shooting_Star(10,random.randrange(480),random.randrange(5) + 5,rand
219                 ropes.append(shooting_star4)
220 
221             #move all the ropes and dots
222             for rope in ropes:
223                 rope.update()
224                 if (rope.x < 0 or rope.x > 640) or (rope.y < 0 or rope.y > 480):
225                     ropes.remove(rope)
226                 if(time_elapsed % 1000 == 0) and time_elapsed != 0:
227                     rope.velocity += 1
228 
229             #if the player is jumping, do not check if it collided with ropes or dots
230             if(octo_cat.immunity == True):
231                 octo_cat.immunity_count += 1
232                 if (octo_cat.immunity_count < OCTO_CAT_JUMP ):
233                     screen.blit(octo_cat.immune_image,(octo_cat.x,octo_cat.y))
234                 else:
235                     octo_cat.immunity = False
236                     octo_cat.immunity_count = 0
237                     screen.blit(octo_cat.image,(octo_cat.x,octo_cat.y))
238             else:
239                 screen.blit(octo_cat.image,(octo_cat.x,octo_cat.y))
240                 for rope in ropes:
241                     if(rope.judge(octo_cat) == True) and (octo_cat.life_lost_time + 30 < time_elapse
242                         octo_cat.life_lost_time = time_elapsed
243                         octo_cat.life -= 1
244                         if octo_cat.life == 0:
245                             endFlag = True
246             if octo_cat.life == 0:
247                 endFlag = True
248                 if pygame.mixer.get_busy():
249                     BGM.stop()  # ゲームオーバー時にBGMを停止
250                     New_BGM.stop()
251                 game_over_sound.play()  # ゲームオーバー音楽を再生
252 
253             for i in range(octo_cat.life - 1):
254                 screen.blit(heart_image,(i * 30,50))
255             pygame.display.update()
256         quit(time_elapsed,force_quit)
257 
258     #when quitting the game
```

```
259     def quit(score,force_quit):
260         if force_quit == False:
261             endFlag = False
262             yourScore = "your score: " + str(score)
263             font1 = pygame.font.SysFont(None, 40)
264             text1 = font1.render(yourScore, False, (255,255,255))
265             font2 = pygame.font.SysFont(None, 40)
266             text2 = font1.render("Press Any Key to Re-Start", False, (255,255,255))
267 
268             while endFlag == False:
269                 screen.fill((0,0,0))
270                 screen.blit(text1,(20,50))
271                 screen.blit(text2,(20,150))
272                 pygame.display.update()
273                 for event in pygame.event.get():
274                     if event.type == pygame.QUIT:
275                         endFlag = True
276                     elif event.type == pygame.KEYDOWN:
277                         endFlag = True
278                         main()
279         #retrieve the highest score
280         data = np.loadtxt("score/score.tsv",dtype="str",delimiter=",")
281         highest_score = data[1]
282         print("The highest score so far: " + highest_score)
283         if(score > int(highest_score)):
284             save_data = np.array([str(datetime.datetime.today()),str(score)])
285             np.savetxt('score/score.tsv',save_data,delimiter=',', fmt="%s")
286         pygame.quit()
287 
288     if __name__ == "__main__":
289         open()
```