UNIVERSIDADE FEDERAL DO AMAPÁ
CIÊNCIA DA COMPUTAÇÃO

EDUARDO LUIGI TAVARES DA SILVA CIUFFI ®

LORENA GOES MONTES ®

LUCAS ALEXANDER DE SOUZA BRITO ®

MATEUS BEZERRA DA SILVA ®

MATHEUS COSTA SILVA ®

eDEATH: documentação oficial ©.

MACAPÁ - AP
2019

1. Introdução

O trabalho proposto tem como objetivo apresentar o game eDEATH. O eDEATH é um projeto desenvolvido como jogo eletrônico inspirado em jogos de fliperama da década de 80, onde o jogador deve coletar pontos na fase e evitar os inimigos para completar a fase. Sua estrutura básica consiste em um jogador principal, que se movimenta pela fase; as bolas vermelhas, itens coletáveis necessários para completar o jogo; os inimigos, os quais se locomovem pela fase e devem ser evitados; e os indicadores visuais informativos na tela.

O jogo possui duração curta, objetivos simples, design minimalista e dificuldade desafiadora.

2. Implementação

- **Personagens**

  O jogo consiste em dois tipo principais de personagens:

  Jogador: personagem principal. É controlado pela pessoa que está jogando.
  Inimigos: personagens que se movem de forma aleatória pela fase. Devem ser evitados pelo jogador.

- **Mecânica**

  A movimentação do personagem no jogo é feita a partir das setas do teclado:
  - **Seta esquerda**: personagem vai para esquerda
  - **Seta direita**: personagem vai para direita
  - **Seta para baixo**: personagem vai para baixo
  - **Seta para cima**: personagem vai para cima

  Os inimigos se movimentam de forma aleatória pelo mapa e devem ser evitados pelo jogador. Se o mesmo encostar nos inimigos, o jogo acaba.

- **Objetivos**

  O jogador deve percorrer o mapa da fase e capturar todos as bolas vermelhas. Não morrer para os inimigos. Feito isso, o jogador chega ao final do game.

- **Fatores de perda**

  O jogo termina caso o jogador encoste em algum dos inimigos que se locomovem pelo mapa.

- **Código Fonte**

  O game foi desenvolvido na linguagem de programação Python, utilizando a biblioteca PyGame. Abaixo foram apresentadas algumas telas do código fonte responsáveis pelas respectivos componentes e funcionalidades no jogo:

1) Criação da Tela (Parede)

```
class Parede(pygame.sprite.Sprite):
    def __init__(self,x,y,largura,altura, cor):
        pygame.sprite.Sprite.__init__(self)

        self.image = pygame.Surface([largura, altura])
        self.image.fill(cor)

        self.rect = self.image.get_rect()
        self.rect.top = y
        self.rect.left = x
```

```
#configuracoes da tela
tela = pygame.display.set_mode([606, 606])

pygame.display.set_caption('Pacman')

background = pygame.Surface(tela.get_size())

background = background.convert()

background.fill(preto)

relogio = pygame.time.Clock()

pygame.font.init()
font = pygame.font.Font("freesansbold.ttf", 24)
```

2) Criação da Moldura

```
#cria as paredes do mapa
def iniciarFaseUm(lista_sprites):
    lista_parede=pygame.sprite.RenderPlain()

    # [x, y, largura, altura]
    paredes = [ [0,0,6,600],
                [0,0,600,6],
                [0,600,606,6],
                [600,0,6,606],
              ]

    for item in paredes:
        parede=Parede(item[0],item[1],item[2],item[3],azul)
        lista_parede.add(parede)
        lista_sprites.add(parede)

    return lista_parede
```

3) Criação da Posição Inicial

```
def iniciarPortao(lista_sprites):
        portao = pygame.sprite.RenderPlain()
        lista_sprites.add(portao)
        return portao
```

4) Criação das Bolas

```
class Bola(pygame.sprite.Sprite):
    def __init__(self, cor, largura, altura):
        # chama o construtor da classe pai (Sprite)
        pygame.sprite.Sprite.__init__(self)

        self.image = pygame.Surface([largura, altura])
        self.image.fill(branco)
        self.image.set_colorkey(branco)
        pygame.draw.ellipse(self.image,cor,[0,0,largura,altura])

        self.rect = self.image.get_rect()
```

5) Jogador (criação, atualização de posição e colisão)

```
class Jogador(pygame.sprite.Sprite):
    mudar_x=0
    mudar_y=0

    def __init__(self,x,y, filename):
        pygame.sprite.Sprite.__init__(self)

        self.image = pygame.image.load(filename).convert()

        self.rect = self.image.get_rect()
        self.rect.top = y
        self.rect.left = x
        self.anterior_x = x
        self.anterior_y = y

    def mudarvelocidade(self,x,y):
        self.mudar_x+=x
        self.mudar_y+=y
```

6) Mudança de direção dos fantasmas

```
class Fantasma(Jogador):
    def mudarvelocidade(self,list,fantasma,virar,passos,l):
      try:
        z=list[virar][2]
        if passos < z:
          self.mudar_x=list[virar][0]
          self.mudar_y=list[virar][1]
          passos+=1
        else:
          if virar < l:
            virar+=1
          elif fantasma == "fantasminha":
            virar = 2
          else:
            virar = 0
          self.mudar_x=list[virar][0]
          self.mudar_y=list[virar][1]
          passos = 0
        return [virar,passos]
      except IndexError:
          return [0,0]
```

```
#definir movimentos dos fantasmas
direcoes = [
  [-30,0,2],
  [0,-15,4],
  [15,0,5],
  [0,15,7],
  [-15,0,11],
  [0,-15,7],
  [-15,0,3],
  [0,15,7],
  [-15,0,7],
  [0,15,15],
  [15,0,15],
  [0,-15,3],
  [-15,0,11],
  [0,-15,7],
  [15,0,3],
  [0,-15,11],
  [15,0,9],
]
```

7) Criação dos Personagens

```
#Localizacao inicial dos personagens
altura_pacman = 301
largura_pacman = 242

largura_fantasma = 400
altura_fantasma = 100

# Adiciona os personagens
Pacman = Jogador( altura_pacman, largura_pacman, "persons/smiley.png" )
lista_sprites.add(Pacman)
colisao_pacman.add(Pacman)

Fantasma1 = Fantasma( altura_fantasma, largura_fantasma, "persons/Clyde.png" )
lista_monstros.add(Fantasma1)
lista_sprites.add(Fantasma1)

Fantasma2 = Fantasma( altura_fantasma, largura_fantasma, "persons/edef.png" )
lista_monstros.add(Fantasma2)
lista_sprites.add(Fantasma2)

Fantasma3 = Fantasma( altura_fantasma, largura_fantasma, "persons/julio.png" )
lista_monstros.add(Fantasma3)
lista_sprites.add(Fantasma3)

Fantasma4 = Fantasma( altura_fantasma, largura_fantasma, "persons/Clyde.png" )
lista_monstros.add(Fantasma4)
lista_sprites.add(Fantasma4)

Fantasma5 = Fantasma( altura_fantasma, largura_fantasma, "persons/Clyde.png" )
lista_monstros.add(Fantasma5)
lista_sprites.add(Fantasma5)
```

8) Chamada para criação e colisão das bolas

```
# Desenha a grade
for linha in range(19):
    for coluna in range(19):
        if (linha == 7 or linha == 8) and (coluna == 8 or coluna == 9 or coluna == 10):
            continue
        else:
          bola = Bola(vermelho, 32, 32)

          # Seta um local aleatorio para o bloco
          bola.rect.x = (30*coluna+6)+26
          bola.rect.y = (30*linha+6)+26

          b_collide = pygame.sprite.spritecollide(bola, lista_parede, False)
          p_collide = pygame.sprite.spritecollide(bola, colisao_pacman, False)
          if b_collide:
            continue
          elif p_collide:
            continue
          else:
            # Adiciona o bloco pra lista de objetos
            lista_bola.add(bola)
            lista_sprites.add(bola)
```

9) Ações do teclado

```
while finalizar == False:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            finalizar=True

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                Pacman.mudarvelocidade(-20,0)
                # Pacman.rotate(90)
            if event.key == pygame.K_RIGHT:
                Pacman.mudarvelocidade(20,0)
            if event.key == pygame.K_UP:
                Pacman.mudarvelocidade(0,-20)
            if event.key == pygame.K_DOWN:
                Pacman.mudarvelocidade(0,20)

        if event.type == pygame.KEYUP:
            if event.key == pygame.K_LEFT:
                Pacman.mudarvelocidade(20,0)
            if event.key == pygame.K_RIGHT:
                Pacman.mudarvelocidade(-20,0)
            if event.key == pygame.K_UP:
                Pacman.mudarvelocidade(0,20)
            if event.key == pygame.K_DOWN:
                Pacman.mudarvelocidade(0,-20)
```

10) Desenho em tela e pontuação

```
if len(blocks_hit_list) > 0:
    pontos +=len(blocks_hit_list)

tela.fill(preto)

wall_list.draw(tela)
portao.draw(tela)
lista_sprites.draw(tela)
lista_monstros.draw(tela)

texto=font.render("Pontos: "+str(pontos)+"/"+str(pontuacao_maxima), True, azul)
tela.blit(texto, [10, 10])
```

11) Resultado Final

```
def resultado(mensagem,left,lista_sprites,lista_bola,lista_monstros,colisao_pacman,lista_parede,portao):
  while True:
      # Processamento de eventos
      for event in pygame.event.get():
        if event.type == pygame.QUIT:
          pygame.quit()
        if event.type == pygame.KEYDOWN:
          if event.key == pygame.K_ESCAPE:
            pygame.quit()
          if event.key == pygame.K_RETURN:
            del lista_sprites
            del lista_bola
            del lista_monstros
            del colisao_pacman
            del lista_parede
            del portao
            iniciarJogo()

      altura_pacman = pygame.Surface((400,200))
      altura_pacman.set_alpha(10)
      altura_pacman.fill((128,128,128))
      tela.blit(altura_pacman, (100,200))

      text1=font.render(mensagem, True, branco)
      tela.blit(text1, [left, 233])

      text2=font.render("Jogar novamente: ENTER.", True, branco)
      tela.blit(text2, [135, 303])
      text3=font.render("Sair: ESC.", True, branco)
      tela.blit(text3, [135, 333])
```

- **Indicadores Visuais**
  O jogo possui três indicadores visuais que auxiliam o usuário a se situar durante a gameplay até seu fim:
    - **Pontuação:** identifica a quantidade de pontos/bolas coletadas pelo usuário. Localizada no canto superior esquerdo.
    - **Game Over**: alerta o jogador com o resultado final do jogo.
    - **Faixas azuis:** delimitação do campo da fase.