Antonio SIlva Martínez A01173663 16 de mayo del 2022

Mod4 act5 ITESM Campus Puebla

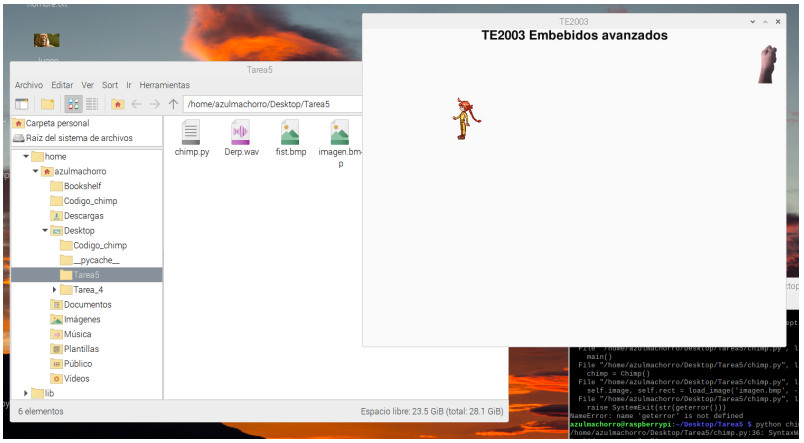
“Realice un programa en python, para que en su raspberry pueda jugar en una animación de 800x600 píxeles, donde el objeto a animar se desplaza a lo largo y ancho de la pantalla de juego. Deberá cambiar el objeto de animación, es decir, no usar la imagen chimp y sustituirla por otra de 54 x 79 píxeles. Además debe cambiar los efectos sonoros del juego. Deberá agregar un icono en el escritorio para su acceso usando un script.”

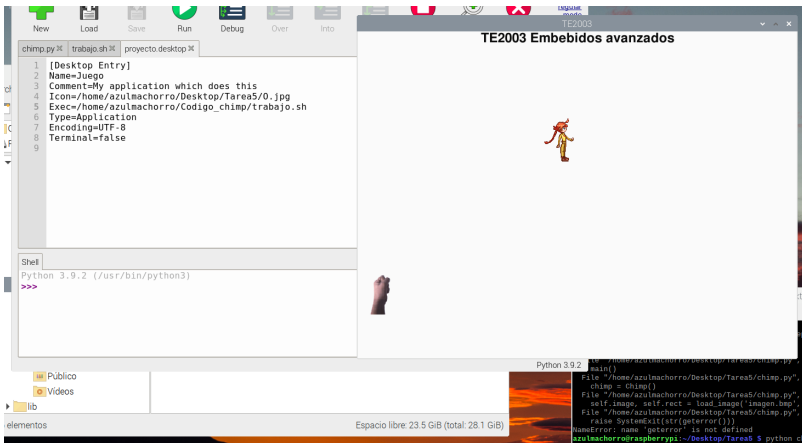
```
#Import Modules
import os, pygame
from pygame.locals import *
import random
#from pygame.compat import geterror

from pygame import mixer

pygame.mixer.init()
if not pygame.font: print ('Warning, fonts disabled')
if not pygame.mixer: print ('Warning, sound disabled')

main_dir = os.path.split(os.path.abspath(__file__))[0]
data_dir = os.path.join(main_dir, 'C:/Users/ansm3/Desktop/Linux_embebido/Modulo_4\chimp')


#functions to create our resources
def load_image(name, colorkey=None):
        fullname = os.path.join(data_dir, name)
        try:
        image = pygame.image.load(fullname)
        except pygame.error:
        print ('Cannot load image:', fullname)
        raise SystemExit(str(geterror()))
        image = image.convert()
        if colorkey is not None:
        if colorkey is -1:
        colorkey = image.get_at((0,0))
        image.set_colorkey(colorkey, RLEACCEL)
        return image, image.get_rect()

#classes for our game objects
class Fist(pygame.sprite.Sprite):
        """moves a clenched fist on the screen, following the mouse"""
        def __init__(self):
        pygame.sprite.Sprite.__init__(self) #call Sprite initializer
        self.image, self.rect = load_image('fist.bmp', -1)
        self.punching = 0

        def update(self):
        "move the fist based on the mouse position"
        pos = pygame.mouse.get_pos()
        self.rect.midtop = pos
        if self.punching:
        self.rect.move_ip(5, 10)

        def punch(self, target):
        "returns true if the fist collides with the target"
        if not self.punching:
        self.punching = 1
        hitbox = self.rect.inflate(-5, -5)
        return hitbox.colliderect(target.rect)

        def unpunch(self):
        "called to pull the fist back"
        self.punching = 0


class Chimp(pygame.sprite.Sprite):
        """moves a monkey critter across the screen. it can spin the
        monkey when it is punched."""
        def __init__(self):
        self.speed = 10         #velocidad de movimiento
        pygame.sprite.Sprite.__init__(self) #call Sprite intializer
        self.image, self.rect = load_image('imagen.bmp', -1)
        screen = pygame.display.get_surface()
        self.area = screen.get_rect()
        self.rect.topleft = 100, 100
        self.move1, self.move2 = self.speed, self.speed
        self.dizzy = 0

        def update(self):
        "walk or spin, depending on the monkeys state"
        if self.dizzy:
        self._spin()
```

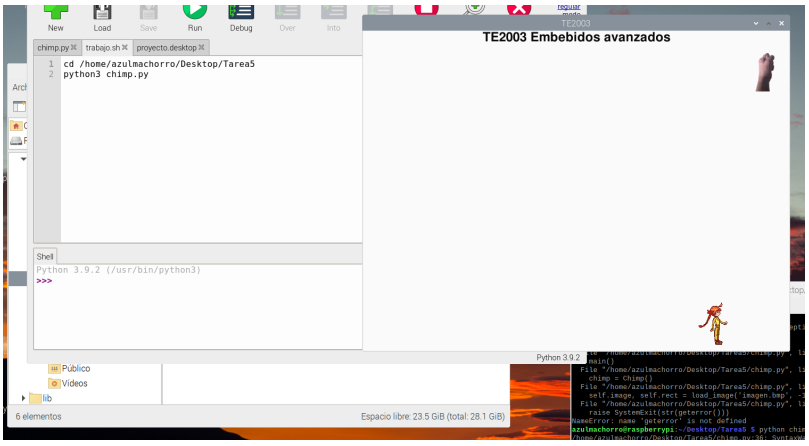
```
                else:
                self._walk()

                def _walk(self):
                "move the monkey across the screen, and turn at the ends"
                newpos = self.rect.move((self.move1, self.move2))

                if self.rect.left < self.area.left or self.rect.right > self.area.right:
                self.move1 = -self.move1
                self.image = pygame.transform.flip(self.image, 1, 0)
                if self.rect.top < self.area.top or self.rect.bottom > self.area.bottom:
                self.move2 = -self.move2

                newpos = self.rect.move((self.move1, self.move2))

                self.rect = newpos

                def _spin(self):
                "spin the monkey image"
                center = self.rect.center
                self.dizzy = self.dizzy + 12
                if self.dizzy >= 360:
                self.dizzy = 0
                self.image = self.original
                else:
                rotate = pygame.transform.rotate
                self.image = rotate(self.original, self.dizzy)
                self.rect = self.image.get_rect(center=center)

                def punched(self):
                "this will cause the monkey to start spinning"
                if not self.dizzy:
                self.dizzy = 1
                self.original = self.image


def main():
                """this function is called when the program starts.
                it initializes everything it needs, then runs in
                a loop until the function returns."""
#Initialize Everything
                pygame.init()
                screen = pygame.display.set_mode((800, 600))
                pygame.display.set_caption('TE2003')
                pygame.mouse.set_visible(0)

#Create The Backgound
                background = pygame.Surface(screen.get_size())
                background = background.convert()
                background.fill((250, 250, 250))

#Put Text On The Background, Centered
                if pygame.font:
                font = pygame.font.Font(None, 36)
                text = font.render("TE2003 Embebidos avanzados", 1, (10, 10, 10))
                textpos = text.get_rect(centerx=background.get_width()/2)
                background.blit(text, textpos)

#Display The Background
                screen.blit(background, (0, 0))
                pygame.display.flip()

#Prepare Game Objects
                clock = pygame.time.Clock()
                PEW_sound = pygame.mixer.Sound('PEW.wav')
                punch_sound = pygame.mixer.Sound('Derp.wav')
                #
                # PEW_sound = pygame.mixer.music.play((r'C:\Users\L03038590\Documents\TE2003\mod4_linux_embebido\whiff.wav'))
                # punch_sound = pygame.mixer.music.play((r'C:\Users\L03038590\Documents\TE2003\mod4_linux_embebido\punch.wav'))
                chimp = Chimp()
                fist = Fist()
                allsprites = pygame.sprite.RenderPlain((fist, chimp))


#Main Loop
                going = True
                while going:
                clock.tick(60)

                #Handle Input Events
                for event in pygame.event.get():
                if event.type == QUIT:
                going = False
                elif event.type == KEYDOWN and event.key == K_ESCAPE:
                going = False
                elif event.type == MOUSEBUTTONDOWN:
```

```
                if fist.punch(chimp):
                punch_sound.play() #punch
                chimp.punched()
                else:
                PEW_sound.play() #miss
                elif event.type == MOUSEBUTTONUP:
                fist.unpunch()

                allsprites.update()

                #Draw Everything
                screen.blit(background, (0, 0))
                allsprites.draw(screen)
                pygame.display.flip()

                pygame.quit()

#Game Over


#this calls the 'main' function when this script is executed
if __name__ == '__main__':
                main()
```

Comprobación de funcionamiento:







Explicación del funcionamiento:

Para el desarrollo de esta actividad hemos utilizado el código “chimp” dado por el profesor en el cual se utilizan las librerías de pygames, siendo que estas nos permite añadir sonido a nuestro programa y nos permite desarrollar la creación de videojuegos, siendo en este caso una especie de “golpea al topo”, siendo que nosotros tendríamos que hacer que un objeto se desplazará a lo largo y de la pantalla para que nosotros pudiéramos golpearlo ocasionando que si fallamos este produzca un sonido y al acertar produjera otro.

Primeramente para el desarrollo de este programa se ha tenido que implementar la librería pygame y la librería random, posteriormente colocamos la ruta en donde se nuestros objetos siendo que está mandará a llamar a estos mismos para que puedan aparecer en la pantalla, posteriormente se crearon dos clases en las cuales se indicarán las acciones de la mano y del objeto al cual tendremos que golpear.

Para la primera clase se creó para el funcionamiento del puño, siendo que este cuenta con cuatro acciones de las cuales se encargan de cargar la imagen, seguir al cursor, hacer el efecto de golpe y regresar a la acción original. Mientras que la segunda clase se encarga de controlar las acciones que tendrá el objeto a golpear, dentro de estas se encuentra el movimiento aleatorio, el dar una vuelta cuando este sea impactado, la velocidad de movimiento y el regresar a la posición de origen después de haber sido golpeado.

Por último el main se encarga de la creación de la superficie de juego, siendo que en este se establece el tamaño, color, y el tamaño de la fuente de algún título que se le quiera agregar, por último dentro del main también se le asignan lo que son las funciones de sonido haciendo que en cado impacto dado o fallido se reproduzca un sonido diferente.

Conclusión:

Para la realización de este trabajo la librería pygames fue de mucha utilidad ya que esta nos permitió realizar algunas acciones de manera más fácil ya que la forma en la que se implementan nos permiten utilizar menos líneas de código haciendo más amigable la forma en la que trabajamos al programar. Esta librería podría ser de utilidad para el desarrollo del reto ya que nos permite el poder manipular los audios de una manera más facil.