UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Programación I

Facundo Falcone - Guevara Mariano - Manuel Samaniego

Biblioteca Pygame 3

Eventos y Colisiones

Eventos

-Los **eventos** en Pygame son interacciones que se generan en respuesta a acciones del usuario. Podemos capturarlos y manejarlos. Hacen el juego interactivo

-**Capturar eventos**

```
while True:
    for event in pygame.event.get():
```

Usamos pygame.event.get() para obtener la lista de eventos que han ocurrido y procesarlos.
Dentro del WHILE con un for each podemos recorrer los eventos 1 por 1

Eventos

```
while True:
    #Obtengo la lista de eventos que estan ocurriendo y la recorro
    for event in pygame.event.get():
        #Verifico el typo del evento para saber si coincide con lo que busco
        if event.type == QUIT:
            #Ejecuto las acciones correspondientes con el evento
            pygame.quit()
            sys.exit()
```

- Haciendo .type a cada evento discernimos cual es.
- El evento del tipo QUIT sirve para cerrar el juego.

Eventos (del teclado)

-Los **eventos del teclado** permiten detectar cuando el usuario presiona o suelta una tecla. Pygame tiene dos eventos principales para el teclado:

. pygame.KEYDOWN: Se activa cuando el usuario presiona una tecla.

. pygame.KEYUP: Se activa cuando el usuario suelta una tecla.

```
if event.type == pygame.KEYDOWN:
    if event.key == pygame.K_LEFT:  # Detecta si se presiona la flecha izquierda
        print("Flecha izquierda presionada")
    if event.key == pygame.K_SPACE:  # Detecta si se presiona la barra espaciadora
        print("Espacio presionado")
```

Eventos (del teclado)

También puedes detectar múltiples teclas a la vez usando pygame.key.get_pressed(), que retorna una lista de todas las teclas presionadas.

```
keys = pygame.key.get_pressed()
if keys[pygame.K_LEFT]:
    mover_izquierda()
if keys[pygame.K_RIGHT]:
    mover_derecha()
```

Eventos (del mouse)

-Al igual que el teclado, Pygame permite capturar eventos del mouse, por ejemplo, si el click fue presionado o soltado

pygame.MOUSEBUTTONDOWN: Cuando se presiona un botón del mouse.

```
if event.type == pygame.MOUSEBUTTONDOWN:
    posicion = (event.pos) #reasigno la tupla posicion al click
    if event.type == pygame.MOUSEBUTTONDOWN:
        if event.button == 1:  # Botón izquierdo
            print("Clic izquierdo detectado")
        elif event.button == 3:  # Botón derecho
            print("Clic derecho detectado")
```

event.pos -> me devuelve una tupla con los valores X, Y donde se hizo click

event.button -> nos devuelve un numérico que representa el botón del mouse donde se hizo click

Eventos Propios

Podemos crear nuestros propios eventos, por ejemplo eventos de tiempo que nos pueden servir para determinadas acciones tales como ejecutar una acción solo cuando se haya cumplido determinado tiempo o controlar los FPS.

Eventos Propios

para ello tenemos que definirlo usando **pygame.USEREVENT**.
Como el **USEREVENT** representa un número entero definido, podemos sumarle un valor entero y tener nuestro propio evento. Como haremos un evento de tiempo, podemos crear una variable con el valor del tiempo en milisegundos para que cada un segundo se ejecute.

```
import pygame as pg

evento_ticks_1 = pg.USEREVENT + 1
tiempo_milisegundos = 1000
```

Eventos Propios

Luego lo que nos queda es asignar nuestro evento al timer de pygame con el método **set_timer()**

```
pg.time.set_timer(evento_ticks_1, tiempo_milisegundos)
```

Esto nos permitirá iterar luego los eventos y verificar si hay algún evento del mismo tipo que el evento que creamos nosotros para poder realizar alguna acción específica como imprimir algún mensaje

```
while True:
    for evento in pg.event.get():
        if evento.type == evento_ticks_1:
            print('Ya paso un segundo...')
```

Facundo Falcone - Guevara Mariano - Manuel Samaniego

Colisiones

La funciones de *collide* se gestionan desde el módulo pygame.Rect, que es un objeto de la librería Pygame y sirve para almacenar coordenadas rectangulares en un programa.

La funciones *collide* se utilizan para observar y reaccionar a las colisiones que se hacen entre dos rectángulos de un programa. Estas 2 funciones *collide* en Pygame implican colisionar al menos dentro de un rectángulo.

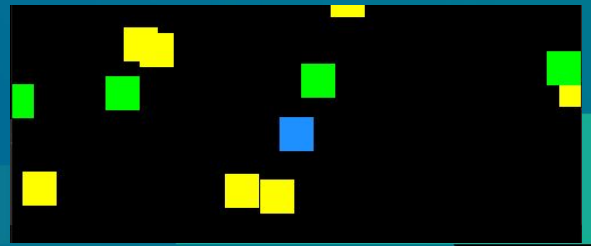
Colisiones

Función pygame.Rect.collidepoint

La función ***pygame.Rect.collidepoint*** permite conocer si un punto ha colisionado dentro de un rectángulo. La función collidepoint (x, y) resulta en un booleano cuando se aplica un rectángulo y, por tanto, pide los puntos para comprobar si colisiona

Función pygame.Rect.colliderect

La función del módulo de Rect de Pygame, **pygame.Rect.colliderect**, comprueba si dos rectángulos se superponen, lo que quiere decir que, si esto llega a pasar, es que han colisionado uno con el otro. Si colisionan, puede ejecutarse un rebote entre ellos. La función colliderect (Rect) resulta en un booleano cuando se aplica a un rectángulo y, además, pide otro rectángulo.

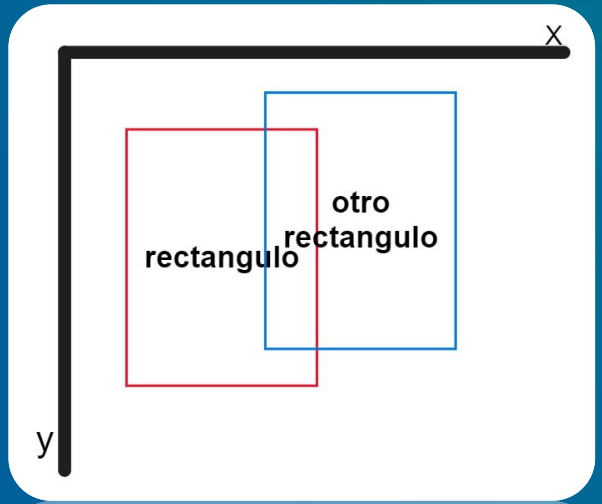


Programación I

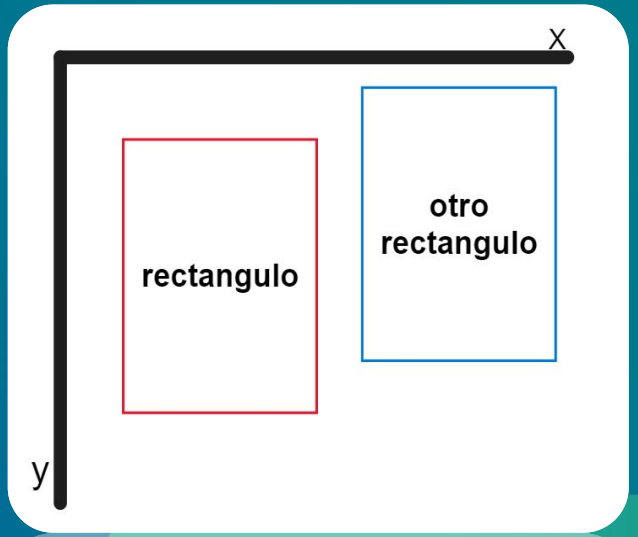
Facundo Falcone - Guevara Mariano - Manuel Samaniego

Biblioteca Pygame 3

```
# Retorna True
rectangulo.colliderect(otro_rectangulo)
```




```
# Retorna False
rectangulo.colliderect(otro_rectangulo)
```



Ejemplo de Colisión y cambio de color

```
while True:
    clock.tick(FPS)
    for evento in pygame.event.get():
        if evento.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    PANTALLA.fill(NEGRO)
    PANTALLA.blit(imagen_vertical, rectangulo_vertical) # -> superpone arriba de pantalla
    PANTALLA.blit(imagen_horizontal, rectangulo_horizontal)

    rectangulo_vertical.y += 10
    if rectangulo_vertical.top > ALTO:
        rectangulo_vertical.bottom = 0

    rectangulo_horizontal.x += 10
    if rectangulo_horizontal.left > LARGO:
        rectangulo_horizontal.right = 0

    if rectangulo_horizontal.colliderect(rectangulo_vertical):
        print("El RECTANGULO COLISIONA y CAMBIA A BLANCO")

        imagen_horizontal.fill(BLANCO)


    else:

        imagen_horizontal.fill(AZUL_CLARO)



    pygame.draw.line(PANTALLA, AZUL, (400,0),(400, 800),1)
    pygame.draw.line(PANTALLA, AZUL, (0,300),(800, 300),1)
    pygame.display.flip()
```