

Pygame (Práctica 3)

Objetivo

Vas a trabajar progresivamente en varios programas hasta conseguir dominar el uso de diferentes elementos en una animación por fotogramas e impedir que éstos salgan de los límites de visualización.

Programa 3: pygame3.py

Parece sencillo pero no lo es. Cuando se ejecuta el programa, con cada click del ratón se dibuja un cuadrado y su color va cambiando con el tiempo...

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# En este programa vamos a generar un número indefinido de cuadrados, uno
# cada vez que el jugador haga click con el ratón. Además, para ilustrar
# los cambios dinámicos, cambiaremos su color al azar en cada fotograma

# Empezamos importando los módulos necesarios

# Fíjate que no importamos directamente pygame.locals como en otras ocasiones
# Eso hace que las variables y constantes se tengan que referenciar de otro
# modo. Por ejemplo, en lugar de QUIT, hay que escribir pygame.QUIT
import pygame, sys

# Queremos también hacer cosas al azar, así que importamos un viejo conocido
# Pero lo hacemos al revés que otras veces. Fíjate que antes teníamos que
# escribir random.randint() y ahora basta poner randint() directamente.
from random import randint

# Creo la surface del programa
visor = pygame.display.set_mode((600,600))

# Guardaremos la lista de cuadrados generados en una lista
cuadrados = []

# En la tupla (x,y) almacenaremos las coordenadas del ratón.
x, y = 0,0

# Vamos al bucle del programa
while True:
    # Primero la lista de eventos
    for evento in pygame.event.get():
        # Si movemos el ratón, actualizamos en consecuencia los valores de x e y
        if evento.type == pygame.MOUSEMOTION:
            x, y = evento.pos
```

```
# Si hacemos click, añadimos un nuevo cuadrado (su centro) a la lista
if evento.type == pygame.MOUSEBUTTONDOWN:
    cuadrados.append((x-5,y-5))
# Para salir del programa, en vez de hacerlo de la manera habitual
# usamos otro evento. Se trata de mirar si el usuario a pulsado
# una tecla (evento KEYDOWN) y si la tecla es la 'q' (K_q)
# se procede a terminar el programa
if evento.type == pygame.KEYDOWN and evento.key == pygame.K_q:
    pygame.quit()
    sys.exit()

# Fuera del bucle de eventos ya, dibujamos el fotograma
# En primer lugar el fondo negro
visor.fill((0,0,0))

# Y en segundo lugar, recorremos la lista de cuadrados y los dibujamos
# uno a uno, eligiendo el color al azar cada vez.
for caja in cuadrados:
    color = (randint(0,255), randint(0,255), randint(0,255))
    pygame.draw.rect(visor, color, (caja, (10,10)))
    pygame.draw.rect(visor, (255,255,255), ((x-5,y-5), (10,10)))

# Terminado. Actualizamos la pantalla volcando el fotograma
pygame.display.flip()
```

Programa 4: cuadradosColor.py

Bien. Modifiquemos el programa anterior para que, durante la animación, los cuadrados mantengan el color con el que se han creado (color que a su vez es aleatorio).

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# cuadradosColor.py

# Este programa es una modificación de pygame3.py en el que el color
# de los cuadrados se deja fijo

import pygame, sys

from random import randint

visor = pygame.display.set_mode((600,600))

cuadrados = [ ]
```

x, y = 0,0

while True:

for evento in pygame.event.get():

if evento.type == pygame.MOUSEMOTION:

x, y = evento.pos

if evento.type == pygame.MOUSEBUTTONDOWN:

PRIMER CAMBIO: El color se decide en la creación del cuadrado
color = (randint(0,255), randint(0,255), randint(0,255))

SEGUNDO CAMBIO: Pongo en la lista el cuadrado y su color
cuadrados.append(((x-5,y-5),color))

if evento.type == pygame.KEYDOWN and evento.key == pygame.K_q:
pygame.quit()
sys.exit()

visor.fill((0,0,0))

for caja in cuadrados:

TERCER CAMBIO: Accedo al cuadrado y su color como elemento de la tupla
pygame.draw.rect(visor, caja[1], (caja[0], (10,10)))
pygame.draw.rect(visor, (255,255,255), ((x-5,y-5), (10,10)))

pygame.display.flip()

Programa 5: cuadradosDiagonales.py

Seguimos modificando. Ahora queremos que los cuadrados no se queden quietos...

#!/usr/bin/env python
-- coding: utf-8 -*-*

cuadradosDiagonales.py

Este programa es una modificación de cuadradosColor.py en el que los cuadrados
se mueven en diagonal hacia la derecha

import pygame, sys

from random import randint

```
visor = pygame.display.set_mode((600,600))

cuadrados = [ ]

x, y = 0,0

while True:

    for evento in pygame.event.get():

        if evento.type == pygame.MOUSEMOTION:
            x, y = evento.pos

        if evento.type == pygame.MOUSEBUTTONDOWN:
            color = (randint(0,255), randint(0,255), randint(0,255))

            # PRIMERA MODIFICACIÓN: Cada cuadrado se añade como una lista y no
            # como una tupla, pues sus elementos (posición) se van a modificar
            cuadrados.append([x-5,y-5,color])

        if evento.type == pygame.KEYDOWN and evento.key == pygame.K_q:
            pygame.quit()
            sys.exit()

    visor.fill((0,0,0))

    for caja in cuadrados:

        # SEGUNDA MODIFICACIÓN: Se modifica la posición de cada cuadrado
        # accediendo a sus coordenadas y sumando 1 a la x y restando 1 a la y
        # Luego, se dibuja normalmente
        caja[0][0] = caja[0][0] + 1
        caja[0][1] = caja[0][1] - 1
        pygame.draw.rect(visor, caja[1], (caja[0], (10,10)))
        pygame.draw.rect(visor, (255,255,255), ((x-5,y-5), (10,10)))

    pygame.display.flip()
```

Programa 6: cuadradosRebotantes.py

Finalmente, nos aseguraremos de que los cuadrados no salgan de la pantalla haciendo que reboten contra sus límites...

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# cuadradosRebotantes.py

# Este programa es una modificación de cuadradosDiagonales.py en el que los
# cuadrados rebotan en los bordes de la ventana

import pygame, sys

from random import randint

visor = pygame.display.set_mode((600,600))

cuadrados = [ ]

x, y = 0,0

# PRIMER CAMBIO:
# Necesitamos almacenar la dirección de movimiento de cada cuadraado.
# Por defecto es sumar 1 a la x (ir hacia la derecha) y restar 1 a la y (ir
# hacia arriba)

dirX = 1
dirY = -1

while True:

    for evento in pygame.event.get():

        if evento.type == pygame.MOUSEMOTION:
            x, y = evento.pos

        if evento.type == pygame.MOUSEBUTTONDOWN:
            color = (randint(0,255), randint(0,255), randint(0,255))

            # SEGUNDO CAMBIO: Para cada cuadrado creado ahora también hay que
            # incluir su movimiento por defecto
            cuadrados.append([x-5,y-5,color, dirX, dirY])

        if evento.type == pygame.KEYDOWN and evento.key == pygame.K_q:
            pygame.quit()
            sys.exit()

visor.fill((0,0,0))
```

for caja in cuadrados:

```
# TERCER CAMBIO: Hay que mirar si el cuadrado rebota.
# Horizontalmente ocurre si la x es 5 o 595 (ya que entonces está
# tocando el borde (recuerda que visor es 600x600)
if caja[0][0] <=5 or caja[0][0] >=595:
    # Para hacer el rebote hay que cambiar el signo correspondiente
    # al dirX del cuadrado
    caja[2] = -caja[2]
# Ahora hay que hacer lo mismo con la dirección vertical:
if caja[0][1] <=5 or caja[0][1] >=595:
    # ... teniendo en cuenta que ahora se actúa sobre la dirY
    caja[3] = -caja[3]

# CUARTO CAMBIO: Ahora el incremento se hace con el almacenado
# en el propio cuadrado.
caja[0][0] = caja[0][0] + caja[2]
caja[0][1] = caja[0][1] + caja[3]
pygame.draw.rect(visor, caja[1], (caja[0], (10,10)))
pygame.draw.rect(visor, (255,255,255), ((x-5,y-5), (10,10)))

pygame.display.flip()
```

Esta vez lo único que se te pide es que escribas cada uno de los programas y compruebes que funcionan. Envíalos todos juntos a tu profesor. Voluntariamente, puedes inventarte un movimiento distinto y dejarte llevar por la imaginación...

Recapitulación

Esta práctica está centrada en que trates de comprender cómo funciona la animación a base de fotogramas y en que manejes las posiciones de los protagonistas mediante coordenadas. Pocos conceptos nuevos aparecen. Pero hay matices...

- ¿Controlas los diferentes tipos de eventos que hemos manejado?
- ¿Hay alguna diferencia entre **update()** y **flip()**?
- ¿Cómo se hace para detectar la pulsación de una tecla?
- Si los elementos de una lista (o una tupla) son, a su vez, otras listas, ¿cómo puede accederse a sus elementos?
- ¿Cómo se hace para que algo, que está moviéndose, rebote?