Universidad Simón Bolívar

Departamento de Computación y Tecnología de la Información

CI-2691: Laboratorio de Algoritmos I

PRIMEROS PASOS CON PYGAME

Autor: Kevin Mena

1. Acerca de Pygame

Pygame es una librería gratis y de uso libre para el lenguaje de programación Python que permite la realización de aplicaciones multimedia, como por ejemplo un videojuego. Es una librería bastante versátil que da una amplia variedad de opciones al programador.

Para descargar la librería solo se debe ir al siguiente link y buscar el sistema operativo en el que se esté trabajando. Dado que usamos Python 3 hay que descargarse la última versión disponible en el siguiente enlace, 1.9.3 Packages (January 16th 2017): https://www.pygame.org/download.shtml

Una forma sencilla también de descargar la librería es abrir el terminal de comandos de cualquiera sea su sistema operativo (con permisos de administrador) y colocar en él el comando:

```
python3 -m pip install -U pygame --user
```

2. Comenzando con Pygame

Primero que nada hay que abrir en su editor de texto preferido el archivo .py en el cual se va a trabajar. Después, antes que cualquier otra cosa, en el código es necesario que se coloquen las siguientes dos líneas al comienzo del archivo luego del encabezado:

```
1   import pygame
2   import os
```

La primera línea importa la librería pygame que se va a utilizar y la segunda línea importa un módulo para funciones del sistema operativo la cual es opcional. Esta servirá más adelante para algunas funcionalidades extra con la ventana, como por ejemplo centrarla cada vez que se abra el programa.

Como segundo paso, comenzaremos declarando algunas constantes que se usarán en nuestro programa. Por convención es una buena práctica colocar las constantes en mayúsculas sostenida, ya que eso ayudará a no confundirnos con alguna variable de nuestro programa. Por tanto, usaremos constantes para establecer el tamaño que tendrá la pantalla, la cual emergerá al iniciar el programa.

```
4   # Variables necesarias para la pantalla
5   ALTO = 720
6   ANCHO = 1280
7   FPS = 30
```

En este caso, se establece que el tamaño será de 1280x720 pero puede ser del tamaño que a ustedes les parezca más adecuado. Segundo, estableceremos una constante a la cual llamaremos "FPS" que nos servirá más adelante para saber la velocidad con la que nuestro programa se actualizará, es decir, a cuántos *frames* o fotogramas por segundo (siendo *frame* un instante o imagen).

Como tercer paso, colocaremos las constantes de los colores que se utilizarán a lo largo del juego ya que para decirle a la librería que color debe utilizar tiene que ser de una forma específica, la cual es el llamado Modelo de color RGB.

- Modelo de color RGB: RGB es un modelo basado en la composición mediante síntesis aditiva de la intensidad de luz de los colores primarios (Rojo, Verde y Azul), siendo posible representar un color por la mezcla de ellos tres, en una tupla con tres valores enteros que representan esa intensidad para cada color. Es decir, añadiendo más o menos de unos de estos tres colores podemos representar cualquier otro color. Por ejemplo, si se quiere representar el Negro se toma 0 de intensidad de los tres colores (0, 0, 0). En la siguiente imagen se observan algunos colores:

```
9   # Colores que seran usados en el juego
10  NEGRO = (0, 0, 0)
11  BLANCO = (255, 255, 255)
12  ROJO = (255, 0, 0)
13  AZUL = (0, 0, 255)
14  VERDE = (0, 255, 0)
```

En la imagen aparecen los colores usados en el ejemplo, por lo que se pueden agregar más de ser necesario.

Ahora que se tienen las constantes con la que se trabajará a lo largo del programa es momento de comenzar a usar la librería y para ello es necesario inicializarla. Esto se hace llamando a la función correspondiente de la librería que sólo es necesario llamar una vez. Luego, se podrá utilizar todas las funciones de la librería y abrir la pantalla. En la imagen aparece la llamada:

```
16  # Inicializar la pantalla del juego
17  pygame.init()
```

A continuación, comenzaremos a configurar algunas opciones de nuestra ventana

```
18  os.environ['SDL_VIDEO_CENTERED'] = '1'              # Centrar la ventana a la hora de abrirse
19  pantalla = pygame.display.set_mode((ANCHO, ALTO))   # Configurando la pantalla
20  pygame.display.set_caption("Mi juego")              # Coloca titulo a nuestra pantalla
21  reloj = pygame.time.Clock()
```

En la línea 18 aparece la función del módulo "os" (opcional), que sirve para centrar la ventana en la pantalla cuanto se abra.

En la línea 19 aparece una variable a la cual llamaremos "pantalla" (para saber qué hace) que tendrá almacenada una referencia al objeto de pygame que contiene todas las funcionalidades para el manejo de la ventana. Usaremos esta referencia para cualquier cambio que queramos hacer en la ventana. La función "pygame.display.set_mode(...)" permite establecer las especificaciones iniciales de la pantalla.

En la línea 20 aparece la función para colocarle título a la ventana.

En la línea 21 del código, la variable llamada "reloj" se inicializa para establecer nuestro tiempo de actualización de la pantalla, usando FPS como se mencionó antes.

Finalizado esto, se tendría el estado inicial de la pantalla.

3. Recomendaciones para usar Pygame en el juego

Dado que el programa a implementar es un juego, es necesario actualizar una y otra vez el estado de la ventana para visualizar las cosas que están ocurriendo. Es obvio asumir que todo lo que pase en nuestro juego estará en un ciclo y que la forma de salir de él dependerá de varios factores, pero se puede resumir en una variable booleana que nos indique si seguimos o no jugando.

```
23   # Loop de juego
24   jugando = True                                  # Variable necesaria para saber si el juego corre
25
26   while jugando:
```

La variable "jugando" se inicializa en True para indicar que se está jugando y se cambia a False en algún momento del ciclo para salir del juego. Una forma de salir es darle click a "cerrar" en la ventana. Para lograr el cierre de la ventana se deben colocar las siguientes instrucciones en el código:

```
while jugando:
    # Como el juego es un loop todo su codigo va AQUI dentro

    # Se verifican los eventos que estan sucediendo
    ''' Siendo eventos definidos como sucesos que ocurren
    dependiendo de ciertas circunstancias (Ej: Tecla presionada)'''
    for evento in pygame.event.get():
        # Si el evento que esta ocurriendo es que se acabo el juego, entonces cerrarlo
        if evento.type == pygame.QUIT:
            jugando = False

# Cerrar el juego
pygame.quit()
```

Como se observa en el código, existen ciertos eventos que pueden generar acciones particulares en el juego. Un evento es un suceso que ocurre bajo circunstancias específicas, como por ejemplo presionar una tecla o un click del mouse sobre cierta área de la pantalla. En el caso del cierre de la ventana, el evento se genera cuando el usuario presiona un click sobre la X localizada en esquina superior derecha de la ventana.

Por tal razón, en el **for** se define la variable llamada "evento" que toma cualquier evento capturado por la función "pygame.event.get()". Esta función devuelve una estructura cuyo campo "type" indica el tipo de evento que ocurrió. En el ejemplo "pygame.QUIT" indica que se cerró la ventana, por lo que se coloca la variable "jugando" en False para salir del loop "**while** jugando", ejecutar la función "pygame.quit()" para cerrar la ventana y dar por terminado el juego.

Luego de todo esto, tendremos listo nuestro programa que genera una pantalla que puede ser cerrada, pero antes veremos que se puede "pintar" la pantalla del color que queramos haciendo uso de la función:

```
# "Pinta" la pantalla de negro
pantalla.fill(NEGRO)
```

Que rellena del color que elijamos la pantalla, usando de referencia nuestra variable pantalla que antes definimos.

Es importante resaltar que como se va a usar el reloj, colocamos dentro de nuestro loop la siguiente llamada

```
# Hacer que el juego corrar a una velocidad que deseemos
reloj.tick(FPS)
```

Esto permitirá configurar la pantalla para que se actualicé según lo que nosotros establecimos (los frames por segundo).

Finalmente la pantalla, al iniciar el programa, quedaría así

4. Algunas funciones útiles para el juego:

Dado que en nuestro juego es necesario crear un tablero y rellenar espacios con círculos se explicarán varias funciones útiles para este propósito:

1) pygame.draw.line(superficie, color, pos_ini, pos_final, intensidad): función que sirve para dibujar una línea. Los parámetros son:
    - Superficie: es en donde se pintara la línea, es decir, nuestra variable "pantalla".
    - Color: el color deseado para dibujar la línea.

- Pos_ini: las coordenadas de la posición inicial desde donde se comenzará a dibujar la línea.
- Pos_final: las coordenadas de la posición final hasta donde se dibujará la línea.
- Intensidad: el grosor en que se dibujará la línea, mínimo 1.

Ejemplo:

pygame.draw.line(pantalla, VERDE, (0,0), (30, 30), 1)

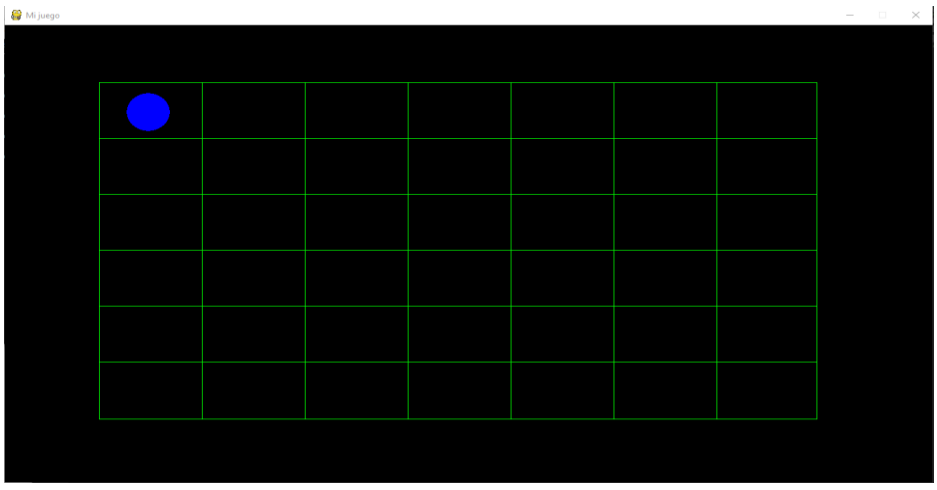
2) pygame.draw.circle(superficie, color, pos, radio, intensidad): función que sirve para dibujar un círculo. Los parámetros son:
   - Superficie es en donde se pintara el círculo, es decir, nuestra variable "pantalla".
   - Color: el color deseado para dibujar el círculo.
   - Pos: la posición del centro del círculo a dibujar.
   - Radio: el radio del círculo a dibujar.
   - Intensidad: que tan marcada será el círculo. El mínimo es 0, que indica que el círculo estará completamente relleno.

Ejemplo:

pygame.draw.circle(pantalla, AZUL, (12,12), 25, 0)

3) pygame.display.flip(): actualiza todas las modificaciones realizadas sobre la pantalla en una iteración del ciclo, lo aplica y lo visualiza. Como el tablero del juego se estará actualizando cada cierto tiempo la mejor forma de hacerlo es de manera oculta al usuario y luego mostrar la pantalla con todas las actualizaciones al final del ciclo. Por tal razón es muy importante: **esta función siempre debe ir en la última línea de su ciclo while.**

Con sólo estas funciones aquí explicadas puedes lograr el siguiente tablero, que es suficiente para visualizar el juego "Cuatro en línea":



Se anexa junto con este documento el archivo de prueba "probandopygame.py" utilizado en los ejemplos mencionados.