Informe proyecto de programación sobre la creación del juego pong en Python

Delgado Sara , Vargas Niedferson.

Universidad Nacional de Colombia, Sede Bogotá

nvargasro@unal.edu.co
sdelgadocu@unal.edu.co

Resumen

El presente informe detalla el desarrollo del proyecto de creación del videojuego 2D "Pong" utilizando el lenguaje de programación Python. El juego se basa en el clásico "Pong", considerado uno de los primeros videojuegos en la historia. Este informe incluye las bibliotecas necesarias para la implementación del juego, las historias de usuario y los requerimientos necesarios.

Abstract

This report details the development of the project for creating the 2D video game 'Pong' using the Python programming language. The game is based on the classic 'Pong,' considered one of the first video games in history. This report includes the necessary libraries for the game's implementation, user stories, and required specifications.

Introducción

En este informe se mostrarán las librerías necesarias para poder realizar el juego “Pong”, se observará la importancia de las librerías *Pygame, Random* y *Sys* en el código del juego. Por otra parte, se mostrarán las historias de usuario y sus correspondientes requerimientos para cumplir sus objetivos.

Bibliotecas Necesarias

El desarrollo del juego requiere el uso de una biblioteca principal y otras herramientas complementarias. A continuación, se describen las principales bibliotecas utilizadas:

Pygame: “Pygame es una biblioteca de Python diseñada para facilitar la creación de videojuegos, especialmente enfocada en juegos 2D, y proporcionando un conjunto de herramientas y funcionalidades que permiten a los desarrolladores gestionar gráficos, sonido y eventos de entrada de manera sencilla y eficaz. Esta biblioteca ha ganado una gran popularidad entre principiantes y educadores debido a su facilidad de uso y la capacidad de enseñar conceptos de programación a través del desarrollo de videojuegos, lo que convierte a Pygame en una excelente opción para introducir a los estudiantes al mundo de la programación y la informática. Además de ser utilizada para fines educativos, Pygame también es utilizada por desarrolladores más experimentados que desean crear prototipos rápidos de juegos o experimentar con ideas de manera ágil, ya que la biblioteca incluye funciones predefinidas que permiten construir proyectos de manera eficiente.

Pygame se utiliza principalmente en tres áreas clave: el desarrollo de videojuegos, el prototipado rápido y la enseñanza de programación. En cuanto al desarrollo de videojuegos, Pygame permite crear juegos 2D completos mediante la gestión de gráficos, animaciones, sonido y entrada del usuario, todo en un entorno de programación sencillo y accesible. Los desarrolladores pueden centrarse en la lógica del juego sin preocuparse demasiado por los detalles de

bajo nivel, ya que la biblioteca abstrae muchos de estos aspectos técnicos. Además, Pygame es ideal para el prototipado rápido, ya que su simplicidad y las funciones prediseñadas permiten a los desarrolladores crear versiones iniciales de sus juegos de manera veloz, facilitando la iteración de ideas y ajustes. Este enfoque es especialmente útil cuando se desea experimentar con un concepto de juego o probar diferentes mecánicas sin tener que dedicar mucho tiempo a la implementación de funciones complicadas. Finalmente, Pygame es ampliamente utilizado en el ámbito educativo, ya que su naturaleza visual y práctica permite enseñar conceptos de programación como bucles, condicionales, manejo de eventos y control de flujo de manera muy accesible. Los estudiantes pueden ver y experimentar de forma inmediata los efectos de su código, lo que facilita el aprendizaje y hace que la programación sea más divertida y menos intimidante.

La biblioteca Pygame incluye una variedad de módulos que facilitan el desarrollo de juegos y la creación de aplicaciones multimedia. El módulo de gráficos es uno de los más importantes, ya que permite cargar imágenes, dibujar formas y manejar la pantalla del juego. A través de este módulo, es posible crear fondos, personajes, objetos interactivos y efectos visuales que forman la base de un juego. También se incluyen herramientas para gestionar animaciones, lo que permite crear secuencias de imágenes para representar movimiento de manera fluida. Por otro lado, el módulo de sonido facilita la incorporación de efectos de sonido y música de fondo, lo que ayuda a mejorar la experiencia del jugador y hacer el juego más inmersivo. Con este módulo, los desarrolladores pueden cargar archivos de audio en diferentes formatos y reproducir sonidos cuando ocurren ciertos eventos en el juego, como el disparo de un arma, la recolección de un objeto o la victoria en una ronda.

El módulo de eventos es otro componente clave de Pygame, ya que se encarga de gestionar las interacciones del usuario con el juego, como las teclas presionadas, los movimientos del ratón y otros tipos de entrada. Este módulo permite que el juego responda a las acciones del jugador de manera dinámica y en tiempo real, lo que es esencial para la jugabilidad. Los desarrolladores pueden programar acciones específicas que ocurran cuando el usuario presiona una tecla o mueve el ratón, lo que abre un abanico de posibilidades para crear controles y mecánicas de juego personalizadas. Finalmente, el módulo de temporización ayuda a controlar la velocidad del juego y sincronizar eventos, lo cual es crucial para mantener una jugabilidad fluida y coherente. A través de este módulo, es posible gestionar los intervalos de tiempo entre eventos y ajustar la velocidad de las animaciones, los movimientos y las acciones del juego.

En resumen, Pygame es una herramienta poderosa y accesible tanto para principiantes como para desarrolladores experimentados, que permite crear juegos 2D de manera rápida y eficiente. Con sus diversos módulos y funcionalidades, facilita el manejo de gráficos, sonido, entrada del usuario y temporización, lo que convierte a Pygame en una opción ideal para el desarrollo de videojuegos, el prototipado rápido y la enseñanza de programación. Además, su comunidad activa y extensa documentación hacen de Pygame una de las bibliotecas más populares y utilizadas en el mundo de la programación de juegos” (KeepCoding, s.f).

Random: El módulo random en Python es una biblioteca estándar que proporciona funciones para generar números pseudoaleatorios y realizar diversas operaciones relacionadas con la aleatoriedad. Este módulo es útil en una amplia gama de aplicaciones, como simulaciones, juegos y cualquier situación donde se necesite un comportamiento aleatorio. Gracias a su simplicidad y versatilidad, el módulo random se utiliza con frecuencia tanto por desarrolladores como por científicos de datos, economistas, educadores y muchos otros

profesionales para agregar un elemento de incertidumbre o azar a sus programas.

Uso del módulo random

El módulo random se utiliza principalmente para:

Generación de números aleatorios: Permite crear números aleatorios en diferentes rangos y distribuciones, lo que es esencial para aplicaciones que requieren variabilidad y aleatoriedad. Por ejemplo, en simulaciones de procesos estocásticos, en juegos de azar o cuando se necesita probar comportamientos en situaciones inciertas.

Selección aleatoria: Facilita la selección aleatoria de elementos de una lista o secuencia, lo que es especialmente útil para juegos o tareas que requieren la elección de un ítem de un conjunto sin un patrón predecible.

Simulaciones: Es muy útil en simulaciones donde se requiere un comportamiento aleatorio, como en juegos de simulación, en experimentos estadísticos o en modelos probabilísticos. Esto incluye la simulación de lanzamientos de dados, cartas o eventos que dependen de probabilidades.

Funciones principales del módulo random

Algunas de las funciones más comunes que ofrece el módulo random incluyen:

random.random(): Devuelve un número de punto flotante aleatorio en el rango [0.0, 1.0), es decir, un valor mayor o igual a 0.0 y menor que 1.0. Esta función es útil cuando se necesita un valor aleatorio en este rango específico para cálculos probabilísticos o normalización de datos.

random.randint(a, b): Devuelve un número entero aleatorio entre los valores a y b, ambos inclusive. Esta función es ideal cuando se requiere un número entero aleatorio dentro de un rango específico, como en la simulación de tiradas de dados o en la selección aleatoria de elementos de un conjunto discreto.

random.choice(sequence): Selecciona un elemento aleatorio de una secuencia no vacía (como una lista o una tupla). Esta función es útil cuando se necesita seleccionar un ítem aleatorio de un conjunto, como elegir un jugador de un equipo o una carta de un mazo.

random.shuffle(x): Mezcla aleatoriamente los elementos de una lista x in situ. Es útil cuando se desea alterar el orden de los elementos en una lista, por ejemplo, para barajar un mazo de cartas, mezclar preguntas en un cuestionario o realizar sorteos aleatorios.

Además de estas funciones principales, el módulo random ofrece muchas otras funciones para generar números aleatorios según distintas distribuciones (como la distribución normal, exponencial, etc.) y para realizar operaciones más avanzadas relacionadas con el azar.

En resumen, el módulo random es una herramienta poderosa y flexible para trabajar con aleatoriedad en Python. Gracias a sus diversas funciones, permite generar números aleatorios, seleccionar elementos de forma impredecible y realizar simulaciones con facilidad, lo que lo convierte en una biblioteca esencial para diversas aplicaciones de programación

Sys: El módulo sys es una biblioteca estándar de Python que proporciona acceso a algunas variables y funciones que interactúan directamente con el intérprete de Python. Este módulo es fundamental para la gestión de la ejecución de programas y facilita diversas operaciones relacionadas con el entorno en el que se ejecuta un programa, lo que lo convierte en una herramienta imprescindible para los desarrolladores que desean personalizar o interactuar con el sistema de ejecución de Python. Entre sus funcionalidades más destacadas, sys permite a los programadores obtener información sobre

el entorno de ejecución, gestionar los argumentos de línea de comandos, manipular la entrada y salida estándar, y acceder a otros parámetros importantes del sistema que afectan a cómo se ejecutan los programas. De esta forma, el módulo sys facilita la personalización del comportamiento de un programa según el entorno, la versión de Python en uso, y las configuraciones del sistema operativo, lo que puede ser muy útil tanto para el desarrollo de aplicaciones como para la depuración y administración de procesos.

El módulo sys se utiliza principalmente en tres áreas clave. En primer lugar, proporciona acceso a los argumentos de línea de comandos, permitiendo que los programas reciban parámetros al ser ejecutados desde la línea de comandos. Esto es esencial cuando se desean pasar configuraciones o datos al programa sin necesidad de modificar su código fuente. A través de los argumentos de línea de comandos, los usuarios o scripts pueden especificar cómo debe ejecutarse el programa, lo cual es común en aplicaciones de consola o cuando se ejecutan scripts automatizados. En segundo lugar, sys facilita la gestión de la salida estándar y de errores, permitiendo redirigir la salida de los programas y gestionar errores de manera más flexible. Esto es especialmente útil para crear aplicaciones robustas, ya que permite definir cómo se deben manejar los errores, registrar información de depuración, o redirigir la salida a archivos o a otros destinos, como puede ser una base de datos o una interfaz de usuario. Por último, el módulo sys proporciona información sobre el entorno de ejecución, lo que permite a los desarrolladores conocer detalles importantes como la versión de Python que se está utilizando, las rutas donde se encuentran los módulos, el sistema operativo en el que se ejecuta el código, y otras variables relacionadas con la configuración del sistema. Esta información puede ser utilizada para hacer que el programa se adapte a diferentes configuraciones, facilitando la portabilidad y la interoperabilidad entre diferentes sistemas y versiones de Python.

Entre las funciones y variables más comunes que ofrece el módulo sys, una de las más útiles es sys.argv. Esta es una lista que contiene los argumentos de línea de comandos pasados al script. sys.argv[0] contiene el nombre del script que se ejecuta, mientras que los demás elementos de la lista corresponden a los argumentos adicionales proporcionados por el usuario o el sistema. Esta funcionalidad permite a los programas procesar información externa de manera flexible y dinámica, y es especialmente útil cuando se requieren configuraciones o parámetros específicos para que el programa funcione correctamente. Otra función clave del módulo sys es sys.exit([arg]), que termina la ejecución del programa. Si se proporciona un argumento, este se utiliza como el código de salida del programa, lo que es útil para indicar si el programa terminó correctamente o si ocurrió un error. Por ejemplo, un código de salida 0 generalmente indica una ejecución exitosa, mientras que otros códigos indican diferentes tipos de errores o fallos. Además, sys.version es una variable que contiene una cadena con la versión de Python que se está utilizando, lo que resulta útil para asegurarse de que el programa se ejecute en la versión correcta de Python y para realizar comprobaciones de compatibilidad entre diferentes versiones del lenguaje. Asimismo, sys.path es una lista de cadenas que especifica las rutas donde Python busca los módulos a importar. Esto permite modificar el comportamiento del programa al incluir directorios personalizados o módulos externos que no estén en la ruta de búsqueda estándar, lo que es útil cuando se trabaja con bibliotecas externas o cuando se quiere modificar la forma en que Python encuentra los módulos.

Además de estas funciones principales, el módulo sys proporciona otras características avanzadas que permiten a los desarrolladores gestionar aspectos más complejos de la ejecución de programas en Python. Por ejemplo, sys.stdin, sys.stdout y sys.stderr

permiten acceder a las streams estándar de entrada, salida y error, lo que es útil para redirigir el flujo de datos o para crear interfaces más complejas que interactúan con el usuario o con otros programas. También es posible utilizar sys.getsizeof() para obtener el tamaño de los objetos en memoria, lo que puede ser útil para optimizar el uso de memoria de los programas.

En resumen, el módulo sys es una herramienta poderosa y fundamental para trabajar con el entorno de ejecución de Python. Proporciona acceso a información clave sobre el sistema, facilita la gestión de la entrada y salida estándar, y permite la personalización de la ejecución del programa mediante argumentos de línea de comandos, todo lo cual contribuye a hacer que los programas sean más flexibles, robustos y adaptables a diferentes entornos. Su uso es esencial para desarrolladores que necesiten controlar el comportamiento de sus programas de manera más precisa y obtener información detallada sobre el sistema y la ejecución del código.

Numpy: Numpy es una librería bastante popular gracias a su funcionamiento con respecto a las físicas, y su profundización en matrices. La definición de Numpy se puede encontrar en su página oficial "*NumPy es el paquete fundamental para la computación científica en Python. Es un Python que proporciona un objeto de matriz multidimensional, varios objetos (como matrices enmascaradas y matrices), y una variedad de rutinas para operaciones rápidas en matrices, incluidas matemáticas, lógicas, manipulación de formas, clasificación, selección, E/S, transformadas discretas de Fourier, álgebra lineal básica, operaciones estadísticas básicas, simulación aleatoria y mucho más.*" (Numpy, 2024). El objeto que representa a Numpy es "***ndarray***" ya que, permite hacer operaciones vectoriales.

Las características que hacen parte de NumPy son**:**

Arrays multidimensionales: Un array multidimensional es una estructura de datos que permite almacenar elementos en más de una dimensión, como en una cuadrícula (2D) o un cubo (3D). Los arrays multidimensionales se utilizan principalmente para el análisis de datos y simulaciones científicas.

```python
import numpy as np

# Crear un array de una dimensión
array_1d = np.array([1, 2, 3, 4, 5])
print(f'Array de una dimensión: {array_1d}')
# Salida: Array de una dimensión: [1 2 3 4 5]

# Crear un array de dos dimensiones
array_2d = np.array([[1, 2, 3], [4, 5, 6]])
print(f'Array de dos dimensiones:\n{array_2d}')
# Salida:
# Array de dos dimensiones:
# [[1 2 3]
#  [4 5 6]]
```

Figura 1 , arrays de primera y segunda dimensión.

```python
# Generar números aleatorios en un array de 3x3
random_array = np.random.rand(3, 3)
print(f'Array de números aleatorios:\n{random_array}')
# Salida:
# Array de números aleatorios:
# [[0.27326464 0.93450999 0.0612561 ]
#  [0.58229708 0.70479523 0.13915861]
#  [0.48986945 0.66327185 0.38387647]]
```

Figura 2 , array con números aleatorios.

SoundFile: En la página oficial se encuentra la documentación que muestra la siguiente documentación, **"** *El SoundFile puede leer y escribir archivos de sonido. La lectura/escritura de archivos es soportado a través de libsndfile, que es una biblioteca gratuita, multiplataforma y de código abierto (LGPL) para leer y escribir muchos formatos de archivos de sonido muestreados diferentes que se ejecutan en muchos plataformas como Windows, OS X y Unix."*(SoundFile, 2022). SoundFile es una librería que permite importar al código (en este caso al juego) música almacenada en los archivo del dispositivos, tener la posibilidad de ambientar el juego con música diferente a la predeterminada por Pygame, puede ofrecer una experiencia diferente a la de otros desarrolladores.

A continuación mostraremos un ejemplo del funcionamiento en código de la librería.

```python
import soundfile as sf
import numpy as np

# Leer un archivo de audio
data, samplerate = sf.read('mi_audio.wav')
print(f'Tasa de muestreo: {samplerate}')
print(f'Datos de audio:\n{data}')

# Modificar los datos de audio (por ejemplo, duplicar el volumen)
data_modificada = data * 2

# Guardar el audio modificado en un nuevo archivo
sf.write('mi_audio_modificado.wav', data_modificada, samplerate)
print('El archivo de audio modificado ha sido guardado.')
```

Figura 3, Demostración SoundFile

Tkinter: Tkinter es la biblioteca estándar de Python para crear interfaces gráficas de usuario (GUIs). Con Tkinter se pueden realizar botones, cuadros de textos, etiquetas y todo lo relacionado con la interfaz. Gracias a esta biblioteca, se puede obtener la capacidad de personalizar el aspecto de los menús. un punto positivo de Tkinter es la capacidad de ser multiplataforma, por ejemplo, se puede utilizar el mismo código tanto en windows como en Mac. La documentación de la biblioteca la encontraremos en la página oficial de Python que dice lo siguiente: “El paquete tkinter («interfaz Tk») es la interfaz por defecto de Python para el kit de herramientas de GUI Tk. Tanto Tk como tkinter están disponibles en la mayoría de las plataformas Unix, así como en sistemas Windows” (Python, 2024).

A continuación se mostrará alguno de los códigos que se utiliza para diferentes propósitos:

- **Crear la Ventana Principal**:

  root = tk.Tk() crea la ventana principal de la aplicación.

  root.title(" ") establece el título de la ventana.

- **Crear una Etiqueta**:

  etiqueta = tk.Label(root, text=" ") crea una etiqueta.

  etiqueta.pack(padx=20, pady=20) añade la etiqueta a la ventana principal y la coloca con un poco de espacio alrededor.

- **Crear un Botón**:

  boton = tk.Button(root, text=" ", command=lambda: print("")) crea un botón con el texto y asocia una acción que imprime un mensaje en la consola al hacer clic.

  boton.pack(padx=20, pady=20) añade el botón a la ventana principal y lo coloca con un poco de espacio alrededor.

- **Iniciar el Bucle Principal**:

  root.mainloop() inicia el bucle de eventos de Tkinter, que espera y maneja eventos como clics de ratón y presiones de teclas.

Historias de usuario y requerimientos

Historias de usuario	Requerimientos
1. Como jugador, quiero poder mover mi paleta hacia arriba y hacia abajo utilizando las teclas de flecha, para poder controlar el movimiento de mi paleta y golpear la pelota.	Establecer un título y un icono para la ventana del juego.
	Permitir el movimiento de las paletas hacia arriba y hacia abajo utilizando las teclas de flecha.
	Limitar el movimiento de las paletas dentro de los límites de la ventana.
2.Como jugador, quiero ver un marcador que muestre los puntos de cada jugador, para poder seguir el progreso del juego	Crear un sistema de puntuación que se muestre en la ventana
	Incrementar la puntuación cuando un jugador anota un punto

3.Como jugador, quiero que la pelota rebote en los bordes superior e inferior de la ventana, para que el juego se sienta más realista.	Hacer que la pelota rebote en los bordes superior e inferior de la ventana.
4.Como jugador, quiero que la pelota rebote en las paletas con un ángulo apropiado, para que el juego sea más desafiante y emocionante	Detectar cuando la pelota toca una paleta y cambiar la dirección de la pelota
	Implementar la lógica para que la pelota rebote en las paletas con un ángulo apropiado
5.Como jugador, quiero poder reiniciar el juego después de que se haya completado una partida, para poder jugar de nuevo.	Permitir que el usuario reinicie el juego después de que se haya completado una partida.
6.Como jugador, quiero poder ajustar la velocidad de la pelota y la sensibilidad de las paletas, para poder personalizar la dificultad del juego.	Agregar una opción de ajuste de la velocidad de las paletas.
	Permitir que el usuario ajuste la velocidad de la pelota y la sensibilidad de las paletas.
7.Como jugador, quiero poder seleccionar el modo de juego (un jugador, dos jugadores, etc.), para poder jugar solo o con amigos.	Añadir una pantalla de inicio con instrucciones y opciones de configuración.
	Permitir que el usuario seleccione el modo de juego (un jugador, dos jugadores, etc.)
8.Como jugador, quiero poder personalizar los colores y la apariencia de los elementos del juego, para poder hacer que el juego se vea más a mi gusto.	Implementar la lógica para determinar cuándo un jugador gana el juego (por ejemplo, el primero en llegar a 10 puntos).
	Permitir que el usuario personalice los colores y la apariencia de los elementos del juego
9.Como jugador, quiero poder ver y comparar mis puntuaciones más altas, para poder seguir mi progreso y mejorar.	Implementar un sistema de puntuación de alta puntuación que se guarde en un archivo.
	Permitir que el usuario vea y compare sus puntuaciones más altas.
10.Como jugador, quiero poder jugar contra un oponente en línea, para poder competir con otros jugadores de todo el mundo.	Implementar la posibilidad de jugar contra un oponente en línea.
11.Como jugador, quiero poder personalizar los controles del juego, para poder jugar de la manera que más me guste.	Permitir que el usuario personalice los controles del juego.
12.Como jugador, quiero poder guardar y cargar partidas, para poder retomar el juego más	Permitir que el usuario guarde y cargue partidas

tarde.	
13.Como jugador, quiero poder desbloquear logros o trofeos durante el juego, para poder sentir un mayor sentido de progreso y logro.	Implementar un sistema de logros o desafíos para los jugadores.
	Implementar un sistema de clasificación de jugadores en línea.
14. Como jugador, quiero que el juego tenga varias ventanas, para poder navegar fácilmente entre las diferentes secciones del juego	Debe incluir las opciones para “jugar”, “Configuración” y “Salir”
	Debe permitir al usuario ajustar opciones como volumen, colores de paletas y controles, así mismo debe incluir botones para “Guardar” y “Cancelar”.
	Debe mostrar el puntaje final y opciones para “Reiniciar” o “Salir al menú”.
15. Como jugador, quiero ver efectos visuales cuando se marca un punto, para que sea más emocionante	Debe haber un efecto gráfico que se active cuando un jugador anota
	El efecto debe ser visible durante unos segundos después de que se marca punto
	Los efectos deben incluir animaciones de destellos o cambios de color
16.Como jugador, quiero una pantalla de pausa, para cuando necesito detener el juego	Debe haber un indicador visual que confirme que el juego está en pausa
	La pantalla debe atenuar el fondo del juego para dar claridad a las opciones
	La pantalla debe mostrar un botón de “Continuar” y “Salir al menú”

Conclusiones

La creación de un juego 2D como Pong en Python no solo implica la implementación de gráficos y lógica de juego, sino también una cuidadosa consideración de las necesidades y deseos de los jugadores. Las historias de usuario presentadas reflejan un enfoque centrado en el usuario, asegurando que cada característica del juego esté alineada con las expectativas del jugador.

Referencias

1. KeepCoding. (s.f.). *¿Qué es Pygame?* Recuperado de https://keepcoding.io/blog/que-es-pygame/

2. Python Software Foundation. (s.f.). *random — Generate pseudo-random numbers. Python 3.12.0 documentation.* Recuperado de: https://docs.python.org/3/library/random.htm

3. Python Software Foundation. (s.f.). *sys — System-specific parameters and functions. Python 3.12.0 documentation.* Recuperado de: https://docs.python.org/3/library/sys.htmll

4. Python Software Foundation. (n.d.). Tkinter – Interfaces gráficas de usuario con Tcl/Tk. Python.org. Recuperado el 9 de diciembre de 2024, de https://docs.python.org/es/3.13/library/tkinter.html
5. Python Software Foundation. (n.d.). SoundFile. PyPI. Recuperado el 9 de diciembre de 2024, de https://pypi.org/project/soundfile/
6. *Python Tutorial - Real-World Practice Projects* www.codecademy.com/get-started/fre