¿Acabas de empezar a aprender Python y quieres desarrollar un proyecto interesante después de aprender lo básico.?

Pues esta serie de videos es exactamente para ti, porque en ella vamos a desarrollar un juego con Python.

Si ya conoces las bases y tomaste algún curso introductorio (no importa de qué canal o de qué curso), esta es la serie que estás buscando, porque te dará las nociones necesarias de cómo empezar a estructurar un proyecto en Python.

En esta serie desarrollaremos un juego llamado Buscaminas, un juego para un solo jugador muy divertido y a la vez desafiante.

Veamos rápidamente el juego que aprenderás a desarrollar.

Ese será el juego que construiremos a lo largo de esta serie.

Como puedes ver, tenemos varias casillas en las que podemos hacer clic y abrirlas. El objetivo principal en Buscaminas es no hacer clic en una casilla que tenga una mina debajo.

Por ejemplo, si hago clic aquí, recibo un número. Ese número significa que alrededor de esa casilla hay una mina que no debo tocar.

Puedo adivinar, por ejemplo, que aquí no hay una mina, y allá tampoco, y resulta que tengo razón. Con el botón derecho del ratón puedo marcar esa casilla como una mina, porque si la clicara con el botón izquierdo, perdería la partida.

La idea es ir adivinando las ubicaciones seguras, haciendo clic en ellas, y evitar las que tienen minas.

Si me equivoco, como en este caso, aparece el mensaje “clicaste en una mina” y pierdo el juego.

El objetivo es descubrir todas las casillas que no tienen minas, y así ganar la partida. Eso implica bastante razonamiento, y será divertido enfrentarnos a los retos que este juego trae consigo.

Antes de empezar, te agradecería que le des “me gusta” a este video para ayudar a que llegue a más gente en YouTube. Y si conoces personas que recién están aprendiendo Python y quieren hacer un proyecto chévere, invítalos a ver esta serie: les será muy útil si están en el nivel intermedio entre principiantes y avanzados.

Ahora sí, ¡empecemos!

Antes de programar, necesitas tener Python instalado en tu computadora, además de un IDE que reconozca el intérprete de Python. Yo usaré Python 3.8, pero cualquier versión superior también servirá, como 3.10 o 3.11.

No usaremos muchas librerías externas ni código dependiente de versiones específicas, así que no habrá problema.

Para este juego en 2D usaremos Tkinter, una librería muy práctica que ya viene incluida con Python y que nos permitirá crear la ventana y los elementos gráficos.

Empezaremos importando Tkinter:

```
from tkinter import *
```

Después, instanciamos una ventana:

```
root = Tk()
```

Esto nos da una ventana básica. Para mantenerla abierta hasta que el usuario la cierre, debemos ejecutar:

```
root.mainloop()
```

Este será el esqueleto de nuestro proyecto.

Una convención en proyectos con Tkinter es llamar a la ventana root, para que sea más fácil encontrar soluciones si buscas en Stack Overflow u otros foros.

Ahora, podemos cambiar atributos de la ventana, como su tamaño con el método geometry, o su título con title. También podemos evitar que la ventana sea redimensionable usando resizable(False, False), lo cual facilita trabajar cuando agreguemos botones y marcos.

Incluso podemos cambiar el color de fondo con:

root.configure(bg="black")

A continuación, crearemos frames. Un frame es un contenedor que nos permite dividir la ventana en secciones, por ejemplo: un encabezado arriba, una barra lateral a la izquierda y el área principal para el juego.

Creamos un frame con:

top_frame = Frame(root, bg="red", width=1440, height=180)

top_frame.place(x=0, y=0)

Así podemos ubicarlo en coordenadas específicas de la ventana.

Con la misma lógica, añadimos un left_frame y un center_frame, cada uno con su color provisional para identificarlo. Más adelante todos quedarán en negro, pero de esta forma podemos ver cómo se distribuyen en pantalla.

Ahora, como estamos repitiendo muchos valores numéricos (como 1440, 720, 180), lo mejor es crear un archivo de configuración (settings.py) para almacenar constantes como el ancho y alto de la ventana.

También podemos crear un archivo de utilidades (utils.py) con funciones que calculen porcentajes del ancho o del alto. Por ejemplo, height_prct(25) devolverá el 25% de la altura total de la ventana.

De esta forma evitamos tener valores “quemados” en el código y hacemos que sea más dinámico.

Finalmente, reestructuramos los frames para usar esas funciones y variables, en lugar de números fijos. Esto hace el proyecto más limpio y escalable.

En este primer episodio ya hemos preparado la ventana, los frames y las bases del proyecto.

En el próximo, aprenderemos a crear más elementos interactivos dentro de la ventana para que podamos empezar a tener un juego básico en funcionamiento.

No olvides darle “me gusta” y suscribirte para no perderte los próximos videos.

¡Nos vemos muy pronto!