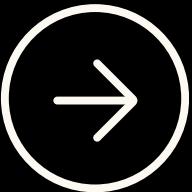


AHORCADO EN PYTHON

VICTOR BEJAR MARTIN



INIDICE



1 Introducion

2 Librerias

3 Base de Datos

4 Inicio

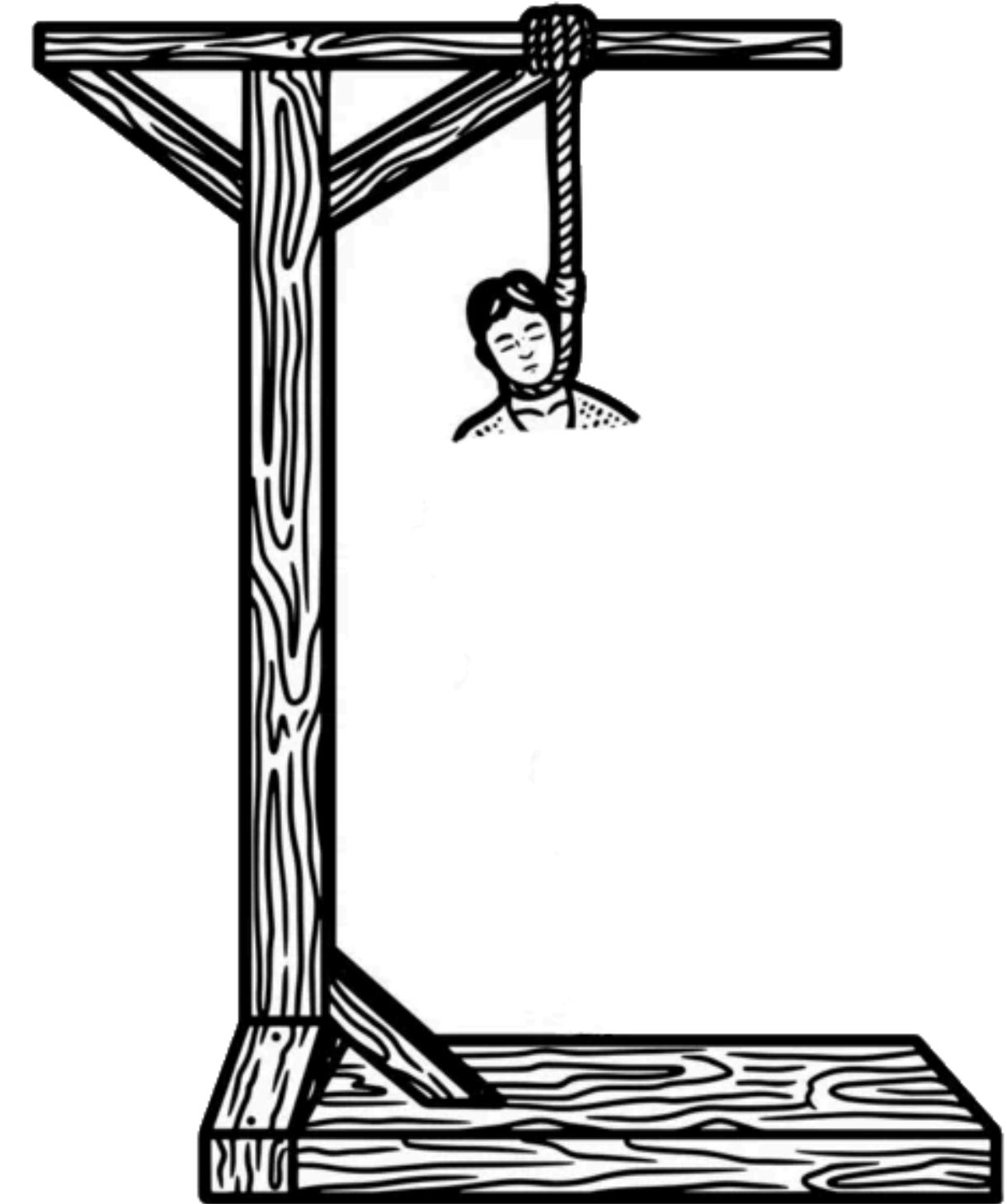
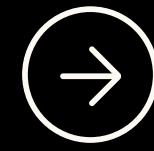
5 Juego

6 Logica

INTRODUCCION

Hemos hecho un proyecto visual en python de un ahorcado.

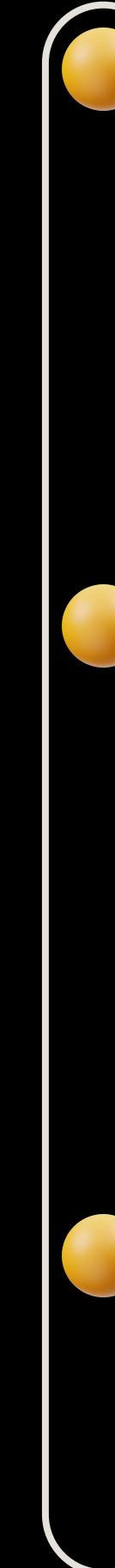
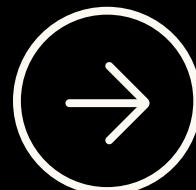
Vamos a ver las cosas usadas en el proyecto. Como librerías, métodos importantes y la BBDD.



LETRA:s



LIBRERIAS



Keyboard

Librería de un lisennner del teclado para que el usuario pueda insertar las letras tecleándolas.

Pillow

Esta librería sirve para manejar imágenes y mostrarlas.

Tkinter

La librería que le da la interfaz al proyecto y con la que mostramos, todo, botones, ventanas, textos...

BASE DE DATOS METODOS

```
def palabras(tipo): 2 usages ▾ victor
    conn = sqlite3.connect('Registro.db')
    cursor = conn.cursor()
    cursor.execute( sql: "SELECT id,palabra FROM palabra WHERE tipo LIKE ?",
                    parameters: (tipo + '%',))
    rel = cursor.fetchall()
    posicion = random.randint( a: 0,len(rel))-1 #valor random de la palabra
    var = rel[posicion] # se coje el valor random de la base de datos
    return var
```

```
def partida(jugador,palabra,ganar): 2 usages ▾ victor
    conn = sqlite3.connect('Registro.db')

    cursor = conn.cursor()
    cursor.execute( sql: "SELECT id FROM jugador WHERE nombre = ?",
                    parameters: (jugador,))
    id = cursor.fetchone()
    cursor.execute( sql: "INSERT INTO partida (ganada,idJugador,idPalabra) VALUES (?,?,?)",
                    parameters: (ganar,id[0],palabra))
    conn.commit()
```

Metodo por el que sacamos palabras aleatorias

Metodo por el cual insertamos las partidas en la BBDD

BASE DE DATOS INICIO

INICIAR

```
def iniciar(): 1 usage  ✅ victor +1
    conn = sqlite3.connect('Registro.db')

    cursor = conn.cursor()

    cursor.execute("CREATE table IF NOT EXISTS jugador"
                  "(id INTEGER PRIMARY KEY AUTOINCREMENT,"
                  "nombre TEXT)")

    cursor.execute("CREATE table IF NOT EXISTS palabra"
                  "(id INTEGER PRIMARY KEY AUTOINCREMENT,"
                  "palabra TEXT,"
                  "tipo TEXT)")

    cursor.execute("CREATE table IF NOT EXISTS partida"
                  "(id INTEGER PRIMARY KEY AUTOINCREMENT,"
                  "ganada BOOLEAN,"
                  "idJugador INTEGER,"
                  "idPalabra INTEGER,"
                  "FOREIGN KEY(idJugador) REFERENCES jugador(id),"
                  "FOREIGN KEY(idPalabra) REFERENCES palabra(id))")

    conn.commit()
```

Cuando no están las tablas creadas las crean sino no hay que cominear para que llegue a la BBDD

DIGITAL MARKETING

```
cursor.execute("SELECT * FROM palabra")
rel = cursor.fetchall()

if len(rel) < 1:
    cursor.execute("")
```

```
# Inserción de datos en la tabla
cursor.executemany( sql: "INSERT INTO palabra (palabra, tipo) VALUES (?, ?)", frutas)
cursor.executemany( sql: "INSERT INTO palabra (palabra, tipo) VALUES (?, ?)", conceptos)
cursor.executemany( sql: "INSERT INTO palabra (palabra, tipo) VALUES (?, ?)", nombres)
conn.commit()
```

Miramos si hay elementos en la tabla palabras sino los metemos hacemos un ejecute y lo comitemos

EL RESTO DE METODOS

Tenemos un par de métodos más pero muy sencillos que solo hacen una consulta

INICIO METODOS

Método por el que seleccionamos la temática

```
def ajustarVentana(ventana, height, width): 1 usage  ± victor
    # Obtener las dimensiones de la pantalla
    screen_width = ventana.winfo_screenwidth()
    screen_height = ventana.winfo_screenheight()

    # Calcular la posición para centrar la ventana
    position_top = int(screen_height / 2 - height / 2)
    position_left = int(screen_width / 2 - width / 2)
```

Método por el cual guardamos los jugadores y los metemos en el juego

```
def guardar(): 1 usage  ± victor
    global inico
    name = nombre.get(index1: "1.0", index2: "end-1c").capitalize()
    if not len(name)==0:
        if not name[0]==": ":
            jugador = database.jugador(name)
            if jugador == None:
                database.insertar(name)
            if palabra != None:
                import Interfaz
                inico.withdraw()
                Interfaz.juego(name,palabra)
```

Método por el cual ponemos la ventana en el medio de la pantalla

```
def tematica(tem): 3 usages  ± victor
    global palabra
    fruta.config(bg="#4449fc")
    informaticos.config(bg="#4449fc")
    nombres.config(bg="#4449fc")
    palabra=tem
    if tem == "fruta":
        fruta.config(bg="red")
    elif tem == "informatica":
        informaticos.config(bg="red")
    elif tem == "nombre":
        nombres.config(bg="red")
```

INICIO METODOS

Método por el cual contabilizamos las partidas ganadas y perdidas

```
def partidas(datos): 1 usage  ± victor
    ganadas = 0
    perdidas = 0
    for list in datos:
        if list[1] == 1:
            ganadas+=1
        else:
            perdidas+=1
    debol = [ganadas,perdid
    return debol
```

Método por el que sacamos los datos del jugador

```
def datos(): 1 usage  ± victor
    global inico, resultado
    name = nombre.get( index1: "1.0" , index2: "end-1c").capitalize()
    if not len(name) == 0:
        if not name[0] == " ":
            jugador = database.jugador(name)
            if jugador == None:
                resultado.config(state=tk.NORMAL)
                resultado.delete( index1: "1.0" , index2: "end")
                resultado.insert(tk.INSERT, chars: "Resultado:\n El usuario no existe en la DB")
                resultado.config(state=tk.DISABLED)
            else:
                datos = database.datos(jugador)
                lista = partidas(datos)
                resultado.config(state=tk.NORMAL)
                resultado.delete( index1: "1.0" , index2: "end")
                resultado.insert(tk.INSERT, chars: f"Resultado:\n GANAR {lista[0]}\n PERDER {lista[1]}")
                resultado.config(state=tk.DISABLED)
```

JUEGO METODOS



Iniciamos fuera de los métodos algunas variables para que no de error

```
ventana = tk.Toplevel()
ventana.resizable( width: False, height: False)
ventana.title("Ahorcado")
frame = tk.Frame(ventana, bg="lightblue")
frame.pack(expand=True, fill="both")
frame.pack_propagate(False)
ajustarVentana(ventana, height: 800, width: 1400)

letraR = tk.StringVar()
palabra = ""
logica = Logica.Logica

img_path = "resorcues/ahorcado7.png"
image = Image.open(img_path)
img_tk = ImageTk.PhotoImage(master= ventana, image=image)
resultado = tk.StringVar()
tematica = ""
letras = tk.StringVar()
```



Método que inicia el juego

```
def juego(jugador,tematicaAux): 1 usage  ↳ victor +1
    global teclas_presionadas, logica, img_tk, resultado, tematica,letras,palabra
    tematica = tematicaAux
    palabra = database.palabras(tematica)
    logica = Logica.Logica(jugador, palabra[1])

    letraR.set("LETRA: _")

    resultado.set(logica.pResultado())

    label1 = tk.Label(frame, textvariable=letraR, font=("Arial", 30), bg="lightblue")
    label1.place(x=390, y=650, height=100, width=200)

    rel = tk.Label(frame, textvariable=resultado, font=("Arial", 30), bg="lightblue")
    rel.place(x=600, y=650, height=100, width=420)

    lAhorcado = tk.Label(frame, image=img_tk)
    lAhorcado.image = img_tk
    lAhorcado.place(x=400, y=20)

    nombre = tk.StringVar()
    nombre.set(logica.jugador)
    lNombre = tk.Label(frame, textvariable=nombre, font=("Arial", 25), bg="lightblue")
    lNombre.place(x=10, y=100)
```

INICIO METODOS

Método por el que reinicia el juego

```
def reinit(): ± victor +1  
    global teclas_presionadas, palabra, logica, tematica, letras  
    palabra = database.palabras(tematica)  
    logica = Logica.Logica(jugador, palabra[1])  
    letraR.set("LETRA: _")  
    letras.set("LETROS:")  
    resultado.set(logica.pResultado())  
    ahorcado()  
    teclas_presionadas = set()
```

Método por el cual cambiamos la imagen del ahorcado

```
def ahorcado(): ± victor  
    global img_tk  
    image = Image.open(f"resorcues/ahorcado{logica.vidas+1}.png")  
    img_tk = ImageTk.PhotoImage(master=ventana, image=image)  
    lAhorcado.config(image=img_tk)
```

Método por el cual ponemos la ventana en el medio de la pantalla

```
def manejar_presion(event): ± victor +1  
    global teclas_presionadas, palabra, logica, letras  
    # Verificar si la tecla es una letra y quedan vidas  
    if event.name.isalpha() and logica.vidas > 0 and not logica.fin():  
        letra = event.name.lower()  
        if letra in abecedario and letra not in teclas_presionadas: #coprobamos  
            teclas_presionadas.add(letra)  
            letraR.set("LETRA:"+letra)  
            if logica.jugar(letra):  
                resultado.set(logica.pResultado())  
                if logica.fin():  
                    win()  
                    database.partida(logica.jugador, palabra[0], ganar: True)  
                else :  
                    ahorcado()  
                    letras.set(f"{letras.get()} {letra}")  
                    if logica.fin():  
                        lose()  
                        database.partida(logica.jugador, palabra[0], ganar: False)
```

INICIO METODOS

Método por el se le dice al jugador si gano y se anima un gif

```
def win(): # victor*
    menu = Toplevel()
    menu.title("GANAR")
    menu.config(bg="lightblue")

    gif_path = "resorcues/win.gif"
    gif = Image.open(gif_path)
    frames = []
    try:
        while True:
            frames.append(ImageTk.PhotoImage(gif.copy()))
            gif.seek(len(frames))
    except EOFError:
        pass

    # Método para actualizar frames

def actualizarGif(frame_index=0): new *
    # Actualizar el frame en el Label
    label.config(image=frames[frame_index])
    label.image = frames[frame_index]

    # Calcular el próximo frame
    next_frame = (frame_index + 1) % len(frames)
    menu.after(ms: 100, actualizarGif, *args: next_frame)

label = tk.Label(menu, bg="lightblue")
label.pack()
tk.Label(menu, text="HAS GANADO", font=("Arial", 20), bg="lightblue").pack()
menu.resizable(width: False, height: False)
ajustarVentana(menu, height: 300, width: 250)
actualizarGif()
```

Método para salir del juego

```
def salir(): # victor
    global ventana, keyboard
    if ventana:
        ventana.destroy()
        ventana = None
    keyboard.unhook_all()
    sys.exit(0)
```

LOGICA METODOS

Método por el que se inicia la lógica

```
def __init__(self, jugador, palabra): 1 usages + victor
    self.jugador = jugador
    self.vidas = 6
    self.palabra = [char.lower() for char in palabra]
    self.resultado = ["_" for _ in palabra]
    self.espacio()
```

Método cada vez que se hace una jugada

```
def jugar(self, letra): 1 usage + victor
    acierto = False
    for i, char in enumerate(self.palabra):
        if letra == char:
            self.resultado[i] = letra
            acierto = True
        if not acierto:
            self.vidas -= 1
    return acierto
```

Método que comprueba el final del juego

```
def fin(self): 3 usages + victor
    if self.resultado == self.palabra:
        return True
    elif self.vidas == 0:
        return True
    return False
```

Método que controla los espacios de la palabra

```
def espacio(self): 1 usage + victor
    for i, char in enumerate(self.palabra):
        if char == " ":
            self.resultado[i] = " "
```

```
def pResultado(self): 3 usages + victor
    aux = ""
    for char in self.resultado:
        aux = "".join([aux, " ", char])
    return aux
```

Método que devuelve el
estado de la palabra en
String

FIN

Gracias por la atención

