import tkinter as tk

import tkinter.messagebox

import time

class Application(tk.Frame):

    def \_\_init\_\_(self, master, \*args, \*\*kwargs):

        tk.Frame.\_\_init\_\_(self, master, \*args, \*\*kwargs)

        self.master = master

        self.running = False

        self.time = 0

        self.hours = 0

        self.mins = 0

        self.secs = 0

        self.build\_interface()

    def build\_interface(self):

        self.time\_entry = tk.Entry(self)

        self.time\_entry.grid(row=0, column=1)

        self.clock = tk.Label(self, text="00:00:00", font=("algerian", 20), width=10)

        self.clock.grid(row=1, column=1, stick="S")

        self.time\_label = tk.Label(self, text="hour   min   sec", font=("algerian", 10), width=15)

        self.time\_label.grid(row=2, column=1, sticky="N")

        self.power\_button = tk.Button(self, text="Start", command=lambda: self.start())

        self.power\_button.grid(row=3, column=0, sticky="NE")

        self.reset\_button = tk.Button(self, text="Reset", command=lambda: self.reset())

        self.reset\_button.grid(row=3, column=1, sticky="NW")

        self.quit\_button = tk.Button(self, text="Quit", command=lambda: self.quit())

        self.quit\_button.grid(row=3, column=3, sticky="NE")

        self.pause\_button = tk.Button(self, text="Pause", command=lambda: self.pause())

        self.pause\_button.grid(row = 3,column=2, sticky = "NE")

        self.master.bind("<Return>", lambda x: self.start())

        self.time\_entry.bind("<Key>", lambda v: self.update())

        self.pause\_button.configure(bg="sky blue")

        self.quit\_button.configure(bg="crimson")

        self.power\_button.configure(bg="green")

        self.reset\_button.configure(bg="gold")

        self.time\_label.configure(bg="light pink")

        self.time\_entry.configure(bg="light yellow")

    def calculate(self):

        """time calculation"""

        self.hours = self.time // 3600

        self.mins = (self.time // 60) % 60

        self.secs = self.time % 60

        return "{:02d}:{:02d}:{:02d}".format(self.hours, self.mins, self.secs)

    def update(self):

        """validation"""

        self.time = int(self.time\_entry.get())

        try:

            self.clock.configure(text=self.calculate())

        except:

            self.clock.configure(text="00:00:00")

    def timer(self):

        """display time"""

        if self.running:

            if self.time <= 0:

                self.clock.configure(text="Time's up!")

            else:

                self.clock.configure(text=self.calculate())

                self.time -= 1

                self.after(1000, self.timer)

    def start(self):

        """start timer"""

        try:

            self.time = int(self.time\_entry.get())

            self.time\_entry.delete(0, 'end')

        except:

            self.time = self.time

        self.power\_button.configure(text="Stop", command=lambda: self.stop())

        self.master.bind("<Return>", lambda x: self.stop())

        self.running = True

        self.timer()

    def stop(self):

        """Stop timer"""

        self.power\_button.configure(text="Start", command=lambda: self.start())

        self.master.bind("<Return>", lambda x: self.start())

        self.running = False

    def reset(self):

        """Resets the timer to 0."""

        self.power\_button.configure(text="Start", command=lambda: self.start())

        self.master.bind("<Return>", lambda x: self.start())

        self.running = False

        self.time = 0

        self.clock["text"] = "00:00:00"

    def quit(self):

        """quit the window"""

        if tk.messagebox.askokcancel("Quit", "Do you want to quit?"):

            root.destroy()

    def pause(self):

        """Pause timer"""

        self.pause\_button.configure(text="Resume", command=lambda: self.resume())

        self.master.bind("<Return>", lambda x: self.resume())

        if self.running == True:

            self.running = False

        self.timer()

    def resume(self):

        """Resume timer"""

        self.pause\_button.configure(text="Pause", command=lambda: self.pause())

        self.master.bind("<Return>", lambda x: self.pause())

        if self.running == False:

            self.running = True

        self.timer()

if \_\_name\_\_ == "\_\_main\_\_":

    """Main loop of timer"""

    root = tk.Tk()

    root.title("TIMER")

    Application(root).pack(side="top", fill="both", expand=True)

    root.mainloop()