import os

import threading

import time

import tkinter.messagebox

from tkinter import \*

from tkinter import filedialog

from tkinter import Tk

from mutagen.mp3 import MP3

from pygame import mixer

root = Tk()

statusbar =Label(root, text="Welcome to Iplayer", relief=SUNKEN, anchor=W, font='Times 10 italic')

statusbar.pack(side=BOTTOM, fill=X)

# Create the menubar

menubar = Menu(root)

root.config(menu=menubar)

# Create the submenu

subMenu = Menu(menubar, tearoff=0)

playlist = []

def browse\_file():

    global filename\_path

    filename\_path = filedialog.askopenfilename()

    add\_to\_playlist(filename\_path)

    mixer.music.queue(filename\_path)

def add\_to\_playlist(filename):

    filename = os.path.basename(filename)

    index = 0

    playlistbox.insert(index, filename)

    playlist.insert(index, filename\_path)

    index += 1

menubar.add\_cascade(label="File", menu=subMenu)

subMenu.add\_command(label="Open", command=browse\_file)

subMenu.add\_command(label="Exit", command=root.destroy)

def about\_us():

    tkinter.messagebox.showinfo('About Iplayer', 'This is a music player build using Python Tkinter Version 1.1')

subMenu = Menu(menubar, tearoff=0)

menubar.add\_cascade(label="Help", menu=subMenu)

subMenu.add\_command(label="About Us", command=about\_us)

mixer.init()  # initializing the mixer

root.title("Iplayer")

leftframe = Frame(root)

leftframe.pack(side=LEFT, padx=30, pady=30)

playlistbox = Listbox(leftframe)

playlistbox.pack()

addBtn = Button(leftframe, text="+ Add", command=browse\_file)

addBtn.pack(side=LEFT)

def del\_song():

    selected\_song = playlistbox.curselection()

    selected\_song = int(selected\_song[0])

    playlistbox.delete(selected\_song)

    playlist.pop(selected\_song)

delBtn = Button(leftframe, text="- Del", command=del\_song)

delBtn.pack(side=LEFT)

rightframe = Frame(root)

rightframe.pack(pady=30)

topframe = Frame(rightframe)

topframe.pack()

lengthlabel = Label(topframe, text='Total Length : --:--')

lengthlabel.pack(pady=5)

currenttimelabel = Label(topframe, text='Current Time : --:--', relief=GROOVE)

currenttimelabel.pack()

def show\_details(play\_song):

    file\_data = os.path.splitext(play\_song)

    if file\_data[1] == '.mp3':

        audio = MP3(play\_song)

        total\_length = audio.info.length

    else:

        a = mixer.Sound(play\_song)

        total\_length = a.get\_length()

    # div - total\_length/60, mod - total\_length % 60

    mins, secs = divmod(total\_length, 60)

    mins = round(mins)

    secs = round(secs)

    timeformat = '{:02d}:{:02d}'.format(mins, secs)

    lengthlabel['text'] = "Total Length" + ' - ' + timeformat

    t1 = threading.Thread(target=start\_count, args=(total\_length,))

    t1.start()

def start\_count(t):

    global paused

    current\_time = 0

    while current\_time <= t and mixer.music.get\_busy():

        if paused:

            continue

        else:

            mins, secs = divmod(current\_time, 60)

            mins = round(mins)

            secs = round(secs)

            timeformat = '{:02d}:{:02d}'.format(mins, secs)

            currenttimelabel['text'] = "Current Time" + ' - ' + timeformat

            time.sleep(1)

            current\_time += 1

def play\_music():

    global paused

    if paused:

        mixer.music.unpause()

        statusbar['text'] = "Music Resumed"

        paused = FALSE

    else:

        try:

            stop\_music()

            time.sleep(1)

            selected\_song = playlistbox.curselection()

            selected\_song = int(selected\_song[0])

            play\_it = playlist[selected\_song]

            mixer.music.load(play\_it)

            mixer.music.play()

            statusbar['text'] = "Playing music" + ' - ' + os.path.basename(play\_it)

            show\_details(play\_it)

        except:

            tkinter.messagebox.showerror('File not found', 'Iplayer could not find the file. Please check again.')

def stop\_music():

    mixer.music.stop()

    statusbar['text'] = "Music Stopped"

paused = FALSE

def pause\_music():

    global paused

    paused = TRUE

    mixer.music.pause()

    statusbar['text'] = "Music Paused"

def rewind\_music():

    play\_music()

    statusbar['text'] = "Music Rewinded"

def set\_vol(val):

    volume = float(val) / 100

    mixer.music.set\_volume(volume)

muted = FALSE

def mute\_music():

    global muted

    if muted:  # Unmute the music

        mixer.music.set\_volume(0.7)

        volumeBtn.configure(text='volume')

        scale.set(70)

        muted = FALSE

    else:  # mute the music

        mixer.music.set\_volume(0)

        volumeBtn.configure(text='mute')

        scale.set(0)

        muted = TRUE

middleframe = Frame(rightframe)

middleframe.pack(pady=30, padx=30)

playBtn = Button(middleframe, text='play', command=play\_music)

playBtn.grid(row=0, column=0, padx=10)

stopBtn = Button(middleframe, text='stop', command=stop\_music)

stopBtn.grid(row=0, column=1, padx=10)

pauseBtn = Button(middleframe, text='pause', command=pause\_music)

pauseBtn.grid(row=0, column=2, padx=10)

# Bottom Frame for volume, rewind, mute etc.

bottomframe = Frame(rightframe)

bottomframe.pack()

rewindBtn = Button(bottomframe, text='rewind', command=rewind\_music)

rewindBtn.grid(row=0, column=0)

volumeBtn = Button(bottomframe, text='volume', command=mute\_music)

volumeBtn.grid(row=0, column=1)

scale = Scale(bottomframe, from\_=0, to=100, orient=HORIZONTAL, command=set\_vol)

scale.set(70)  # implement the default value of scale when music player starts

mixer.music.set\_volume(0.7)

scale.grid(row=0, column=2, pady=15, padx=30)

def on\_closing():

    stop\_music()

    root.destroy()

root.protocol("WM\_DELETE\_WINDOW", on\_closing)

root.mainloop()